PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H03M 13/00, H04L 9/30

A1

(11) International Publication Number: **WO 00/51244**

(43) International Publication Date: 31 August 2000 (31.08.00)

(21) International Application Number: PCT/US00/03522

(22) International Filing Date: 10 February 2000 (10.02.00)

(30) Priority Data:

60/119,674	11 February 1999 (11.02.99)	US
60/137,687	4 June 1999 (04.06.99)	US

(71) Applicant *(for all designated States except US)*: RSA SECURITY INC. [US/US]; 20 Crosby Drive, Bedford, MA 01730 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: JUELS, Ari [US/US]; 131 Freeman Street, Apt. 3, Brookline, MA 02446 (US). WATTENBERG, Martin, M. [US/US]; Apartment 2C, 328 West 19th Street, New York, NY 10011 (US).

(74) Agent: LANZA, John, D.; Testa, Hurwitz & Thibeault, LLP, High Street Tower, 125 High Street, Boston, MA 02110 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

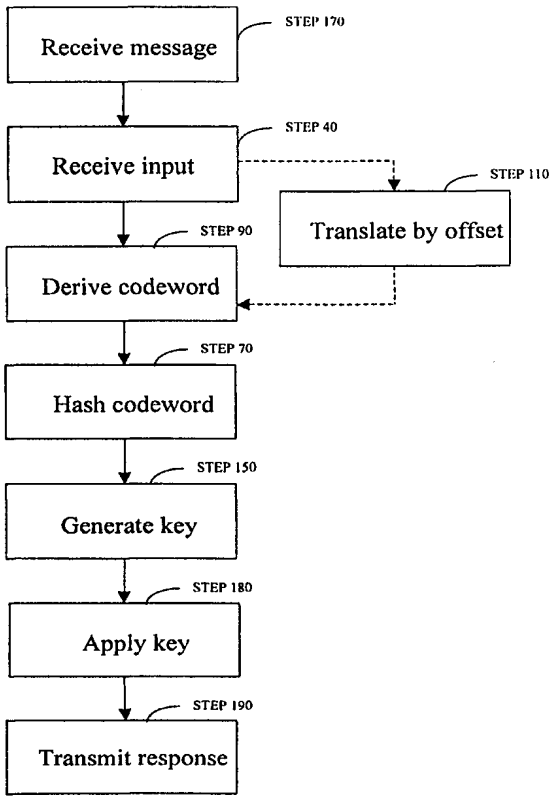
With international search report.

Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: A FUZZY COMMITMENT SCHEME

(57) Abstract

The present invention combines techniques from the areas of error–correcting codes and cryptography to achieve a new type of cryptographic primitive referred to as a fuzzy commitment scheme. Like a conventional cryptographic commitment scheme, a fuzzy commitment scheme is both concealing and binding: it is infeasible for an attacker to learn the committed value, and also for the committer to decommit a value in more than one way. The scheme is fuzzy in the sense that it accepts a witness that is close to the original encrypting witness in a suitable metric, but not necessarily identical. This characteristic of our fuzzy commitment scheme makes it particularly useful for applications such as biometric authentication systems, in which data is subject to random noise. Because the scheme is tolerant of error, it is capable of protecting biometric data just as conventional cryptographic techniques, like hash functions, are used to protect alphanumeric passwords. A fuzzy commitment scheme includes using a decoding function to map an input pattern to a first codeword selected from the plurality of codewords associated with an error–correcting code, calculating an offset between the input pattern and the first codeword, and hashing the first codeword. The hash of the first codeword in association with the offset form a fuzzy commitment.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

A FUZZY COMMITMENT SCHEME

Cross-Reference to Related Application

This application claims priority to and the benefit of U.S. Provisional Patent Application Serial Number 60/119,674, which was filed on February 11, 1999, and U.S. Provisional Patent Application Serial Number 60/137,687, which was filed on June 4, 1999, both of which are incorporated by reference.

Technical Field

The present invention relates to error-correcting codes and to cryptography. In particular, the present invention relates to a combination of cryptography and error-correcting codes to achieve a new cryptographic primitive.

Background Information

Cryptographic protocols are conventionally predicated on exact knowledge. An authentication system using RSA signatures, for example, derives its security largely from the presumption that a legitimate user with public key (N, e) possesses a corresponding secret key of the uniquely specifiable form (N, d). There are situations, however, in which human and other factors undermine the possibility of exactness in a security system. For example, in biometric systems in which users identify themselves by means of fingerprint features, variability in user interaction is such that a finger is rarely read exactly the same way twice. Moreover, there are situations in which although the original information in a system is exact, its transmission may only be approximate. For example, users typically make typing errors when entering passwords on keyboards. Similarly, data transmission channels are often subject to random noise.

An element of some cryptographic protocols is referred to as a bit commitment scheme. In a conventional bit commitment scheme, one player, whom we denote the sender, aims to conceal a bit b. The sender produces an encryption of b, denoted by y, and sends y to a second player, known as the receiver. Generally, a bit commitment scheme is such that it is infeasible for the second player to learn the bit b. Additionally, the sender later "opens" the commitment y, that is, proves to the receiver that y indeed represents an encryption of b. It is generally only feasible, however, for the sender to "open" y in one way, that is, to decrypt a unique value of b.

We may view this, intuitively, as a process whereby the sender places the bit b in a safe and gives the safe to the receiver. Only the sender can open the safe, since she alone knows the combination. Moreover, she cannot change the value contained in the safe while it is in the keeping of the receiver.

An example of a bit commitment scheme is the storage of the hash of user's password in a UNIX file accessible only to the UNIX system administrator. Since the system administrator only has access to the hash of the password, the system administrator does not know what the user's plaintext password is. Nonetheless, when the user provides a password for authentication, the system administrator can compare the hash of the provided password to the stored hash and, if the hashes match, confirm that the user has provided the proper password. Bit commitment may alternatively be done, for example, using a symmetric encryption algorithm, an asymmetric encryption algorithm, a pseudo-random sequence generator, or any other one-way function.

Formally, a bit commitment scheme consists of a function $F: \{0, 1\} \times X \rightarrow Y$. To commit a bit b, the sender chooses a witness $x \in X$, generally uniformly at random. The sender then computes $y = F(b, x)$. This value y is known as a blob. It represents the bit b sealed in a "safe". To "open" or decommit the blob y, the sender produces the bit b and the witness x. The blob is successfully opened if the receiver has been convinced that y indeed represents an encryption of b. A bit commitment scheme is said to be concealing if it is infeasible for the receiver to guess b with probability significantly greater than 1/2. It is said to be binding if it is infeasible for the sender to decommit the blob y with the incorrect bit, that is, with $(1 - b)$. It is possible to deploy a bit commitment scheme as a commitment scheme on an arbitrarily long string of bits by committing each bit independently. The term commitment scheme shall refer to a scheme that involves commitment of a bit string c (or other potentially non-binary value) in a single blob, and for which it is possible to extract c efficiently given a witness for the blob. Thus we assume $F: C \times X \rightarrow Y$, where B is some potentially non-binary space.

Vendors of biometric systems have for some time recognized the importance of achieving a practical system that stores biometric information in a non-explicit, protected form and that also can tolerate some corruption in subsequent biometric readings. To this end, the Mytec Technologies Inc. has developed an encryption process in which biometric information serves as an unlocking key. Sold under the brand name Bioscrypt™, Mytec Technologies's process overcomes the problem of corruption in biometric readings by means of Fourier transforms. While fairly efficient, however, the Bioscrypt™ process carries no rigorous security guarantees.

Davida, Frankel, and Matt have proposed a system in which a biometric template is stored in non-explicit, protected form. The Davida et al. system requires multiple biometric readings from which the check bits may be derived. A hash of the Davida et al. template which includes the check bits is then stored. The multiple biometric readings required by the Davida et al. system may be too time-consuming to be practical or attractive for many real-world applications. Further, the Davida system does not have the necessary error tolerance to work in many real-world applications.

Summary Of The Invention

A simple cryptographic primitive, which is a type of commitment scheme that uses well-known algorithms, facilitates the use of approximate information in cryptographic systems. As a model for approximate reasoning in humans, researchers in artificial intelligence have elaborated a notion known as "fuzzy logic." By analogy, we call the primitive introduced in this application a fuzzy commitment scheme. The fuzzy commitment scheme allows for a stronger notion of binding than that previously employed in the literature. Using the fuzzy commitment scheme, it is not just infeasible to decommit two distinct values from a single commitment, but also, it is infeasible to decommit using two substantially different witnesses.

The fuzzy commitment scheme F is able to achieve a new commitment scheme property referred to as "fuzziness". By this, we mean that the commitment scheme is resilient to small corruptions in witness values. More precisely, a blob y can be opened using any witness x′ that is close to x in some appropriate metric, such as Hamming distance, but not necessarily identical to x. At first glance, having this type of resilience seems contradictory to the goals that F be binding and concealing. After all, to achieve these two security aims, F must be an encryption function of sorts. It would therefore seem necessary, in accordance conventional encryption or hash function design, for small changes in input values to yield large, unpredictable changes in output values. In other words, F should thoroughly and unpredictably "scramble" input bits. On the other hand, the goal of fuzziness in F suggests exactly the opposite, namely a high degree of local structure. Here, we reconcile these ostensibly conflicting goals using well-known components drawn from error-correcting codes and cryptography.

In general, in one aspect, the invention relates to a method for generating a fuzzy commitment from an input pattern. The method includes receiving an input pattern from a user and mapping the input pattern to a first codeword. The first codeword is selected at random from

a plurality of codewords associated with an error-correcting code. The method also includes calculating an offset between the input pattern and the first codeword, and hashing the first codeword.

In one embodiment, the input pattern is a metric associated with a user such as a first measurement of a biometric or a first measurement of a pattern of behavior. In another embodiment, the offset is stored. In another embodiment, both the offset and the hash of the first codeword is stored.

In another embodiment, a key is generated using an encryption algorithm and the hash of the first codeword as the key generation seed. In one embodiment, the key is used to encrypt a message. In a related embodiment, the key is further used to decrypt the encrypted message. In another embodiment, the key is used to decrypt a message without having been previously used to encrypt the message. In another embodiment, the key is used to sign a message.

In another embodiment, a key pair is generated using an asymmetric encryption algorithm and the hash of the first codeword as the key generation seed. The key pair includes a public key and a private key. In a related embodiment, the public key is transmitted to an authentication entity.

In general, in another aspect, the invention also relates to a method for generating a fuzzy commitment from an input pattern. The method includes receiving an input pattern from a user and deriving a first codeword from the input pattern. The first codeword is derived by applying a decoding function of an error-correcting code to the input pattern. The method also includes hashing the first codeword.

In one embodiment, the input pattern is a metric associated with a user such as a first measurement of a biometric or a first measurement of a pattern of behavior. In another embodiment, the hash of the first codeword is stored. In another embodiment, the hash of the first codeword is compared to a stored hash and the input pattern is authenticated when the two hashes match.

In another embodiment, the input pattern is translated by an offset and the first codeword is derived from the translated input. In a related embodiment, the hash of the first codeword is then compared to a stored hash and the input pattern is authenticated when the two hashes match.

In another embodiment, a key is generated using an encryption algorithm and the hash of the first codeword as the key generation seed. In a related embodiment, the key is used to encrypt a message. The key may further be used to decrypt the encrypted message. In a related

embodiment, the key is used to decrypt a message without having been previously used to encrypt the message. The encrypted message that is decrypted may include the offset in an unencrypted form. In another related embodiment, the key is used to sign a message.

In another embodiment, a key pair is generated using an asymmetric encryption algorithm and the hash of the first codeword as the key generation seed. The key pair includes a public key and a private key. In a related embodiment, the public key is transmitted to an authentication entity.

In general, in another aspect, the invention relates to an apparatus for generating a fuzzy commitment from an input pattern. The apparatus includes an input device that receives an input pattern from a user and a mapper that maps the input pattern to a first codeword. The first codeword is selected at random from a plurality of codewords associated with an error-correcting code. The method also includes an offset calculator that calculates an offset between the input pattern and the first codeword, and a hasher that hashes the first codeword.

In one embodiment, the input device accepts a metric associated with a user such as a first measurement of a biometric or a first measurement of a pattern of behavior. In another embodiment, the apparatus also includes a storage device for the offset. In another embodiment, the apparatus includes a storage device for the offset and the hash of the first codeword.

In another embodiment, the apparatus includes a key generator that generates a key using an encryption algorithm and the hash of the first codeword as the key generation seed. In one embodiment, the apparatus also includes an encryption device that encrypts a message with the key. In another embodiment, the apparatus includes a decryption device that decrypts an encrypted message with the key. In another embodiment, the apparatus includes both an encryption device that decrypts an encrypted message with the key and a decryption device that decrypts the encrypted message with the key. In another embodiment, the apparatus includes a signature device that signs a message with the key.

In another embodiment, the apparatus includes a key generator that generates a key pair using an asymmetric encryption algorithm and the hash of the first codeword as the key generation seed. The key pair includes a public key and a private key. In a related embodiment, the apparatus also includes a transmission device that transmits the public key to an authentication entity.

In one embodiment, the apparatus also includes a key generator, an encryption device, and a concatenator. The key generator generates a key pair using an encryption algorithm and the

hash of the first codeword as the key generation seed. The encryption device encrypts a message using the key and the concatenator joins the offset to the encrypted message.

In general, in another aspect, the invention relates to an apparatus for generating a fuzzy commitment from an input pattern. The apparatus includes an input device that receives an input pattern from a user and a codeword generator that produces a first codeword from the input pattern. The first codeword is produced by applying a decoding function of an error-correcting code to the input pattern. The method also includes a hasher that hashes the first codeword.

In one embodiment, the input device accepts a metric associated with a user such as a first measurement of a biometric or a first measurement of a pattern of behavior. In another embodiment, the apparatus includes a storage device for the hash of the first codeword. In another embodiment, the apparatus also includes a comparator that compares the hash of the first codeword to a store hash and authenticates the input pattern when the two hashes match.

In another embodiment, the apparatus includes a translator that translates the input pattern by an offset. The codeword generator in this embodiment produces a codeword from the translated offset. In a related embodiment, the apparatus also includes a comparator that compares the hash of the first codeword to a store hash and authenticates the input pattern when the two hashes match.

In a related embodiment, the apparatus includes a key generator that generates a key using an encryption algorithm and the hash of the first codeword as the key generation seed. In one embodiment, the apparatus also includes an encryption device that encrypts a message with the key. In another embodiment, the apparatus includes a decryption device that decrypts an encrypted message with the key. In a related embodiment, the encrypted message includes the offset in an unencrypted form. In another embodiment, the apparatus includes both an encryption device that decrypts an encrypted message with the key and a decryption device that decrypts the encrypted message with the key. In another embodiment, the apparatus includes a signature device that signs a message with the key.

In another embodiment, the apparatus includes a key generator that generates a key pair using an asymmetric encryption algorithm and the hash of the first codeword as the key generation seed. The key pair includes a public key and a private key. In a related embodiment, the apparatus also includes a transmission device that transmits the public key to an authentication entity.

In general, in one aspect, the invention relates to a method for registering an input pattern.

The method includes receiving an input pattern from a user and mapping the input pattern to a first codeword. The first codeword is selected at random from a plurality of codewords associated with an error-correcting code. The method also includes calculating an offset between the input pattern and the first codeword, hashing the first codeword, and storing the offset and a hash of the first codeword. In one embodiment, the input pattern received includes a first measurement of a biometric associated with a user. In another embodiment, the input pattern includes a first measurement of a pattern of behavior associated with a user.

The foregoing and other objects, aspects, features, and advantages of the invention will become more apparent from the following description and from the claims.

Brief Description Of The Drawings

In the drawings, like reference characters generally refer to the same parts throughout the different views. Also, the drawings are not necessarily to scale, emphasis instead generally being placed upon illustrating the principles of the invention.

FIG. 1 is a block diagram showing an error-correcting code, a portion of which is used in aspects of the invention.

FIG. 2 is an example of decoding with unconstrained codewords.

FIG. 3 is an example of mapping with constrained codewords.

FIG. 4 is an example of decoding with constrained codewords and an offset.

FIGs. 5A and 5b are a comparison of the security levels associated with different numbers of codewords.

FIG. 6 is a functional block diagram of one aspect of a fuzzy commitment.

FIG. 7 is a functional block diagram of another aspect of a fuzzy commitment.

FIG. 8 is a functional block diagram of the registration of an input pattern.

FIG. 9 is a functional block diagram of the registration of an input pattern using unconstrained codewords.

FIG. 10 is a functional block diagram of the authentication of a registered input pattern.

FIG. 11 is a functional block diagram of the registration of an input pattern using a key.

FIG. 12 is a functional block diagram of the registration of an input pattern using a key and unconstrained codewords.

FIG. 13 is a functional block diagram of the authentication of a registered input pattern based on the response to a challenge message using a key.

FIG. 14 is a functional block diagram of the encryption of a message.

FIG. 15 is a functional block diagram of the encryption of a message using unconstrained codewords.

FIG. 16 is a functional block diagram of the decryption of a message using unconstrained codewords.

FIG. 17 is a functional block diagram of the derivation of a key using a hash of a codeword.

FIG. 18 is a functional block diagram of the derivation of a key using a previously calculated offset or unconstrained codewords.

FIG. 19 is an apparatus for registering an input.

Description

Referring to FIG. 1, an error-correcting code is illustrated. A portion of an error-correcting code is used in embodiments of the invention. Generally, an error-correcting code is used to enable transmission of a message intact over a noisy communication channel. A message m to be transmitted is chosen from message space 10. The set of messages M in message space 10 may be represented mathematically as $M = \{0, 1\}^k$ where each message m in the set of messages M is a binary k-bit string. There are 2^k messages in the set of messages M because each bit in the k-bit string can have one of two values.

The message m is provided as input to a translation function g. The translation function g translates the message m into a codeword c in codeword space 20. The translation function g represents a one-to-one mapping of a message m from message space 10 to a codeword c in codeword space 20. Accordingly, for each message m, there is one corresponding codeword c. An error-correcting code for use with a binary set of messages M that are k-bits in length contains a set of codewords C including 2^k codewords since there is one codeword c for each of the 2^k messages. The operation of the translation function g can be described mathematically as $g: M \rightarrow C$. The set of codewords C in codeword space 20 may be described mathematically as $C \subseteq \{0, 1\}^n$ where each codeword c in the set of codewords C is a binary n-bit string. Generally, the message m is different from codeword c at least because codeword c contains redundant elements. If a codeword c contains redundant elements, the length of the codeword c bit string n will be greater than the length of the message m bit string k.

The codeword c is transmitted 30 over a communication channel. Noise 35 may be introduced during transmission 30 so that a corrupted codeword i, which is generally some variation of codeword c, is received at the receiving end of the communication channel. The corrupted codeword i is provided as input to a decoding function f. The decoding function f reconstructs the codeword c from the corrupted codeword i. The redundant elements of the codeword c allow the decoding function to perform this reconstruction.

The decoding function f maps a corrupted codeword i to a codeword c in the set of codewords C. A corrupted codeword i may be an arbitrary n-bit binary string. When the decoding function f is successful, it maps a corrupted codeword i to the nearest codeword c in the set of codewords C. In this context, the nearest codeword c is the codeword c that is the closest by an appropriate metric from the corrupted codeword.

The task of mapping an arbitrary string to its nearest codeword is known as the maximum likelihood decoding problem. Practical classes of codes with polynomial-time solutions to this broad problem are at present unknown. Conventional decoding functions perform a more limited task in that they successfully decode any word that lies within a certain radius of some codeword. Such decoding functions can be used in embodiments described herein.

Generally, when a decoding function f fails, it outputs ϕ. (Some error correcting codes may operate somewhat differently. For example, list decoding functions f yield a set of candidate codewords, rather than a single correct one. The underlying principles remain the same in such settings.) The operation of the decoding function f can be described mathematically as $f: \{0, 1\}^n \rightarrow C \cup \{\phi\}$. The reverse translation function g^{-1} is used upon receipt of a reconstructed codeword c to retrieve the original message m.

The robustness of an error-correcting code depends on the minimum distance of the code. In this description, Hamming distance and Hamming weight will be used as an example of a way to measure the minimum distance of a binary block code. If the Hamming weight of an n-bit binary string u is defined to be the number of '1' bits in u and the Hamming weight of an n-bit string u is denoted by $\| u \|$, then the Hamming distance between two binary bitstrings u and v is defined to be the number of bits in which the two strings differ. The Hamming distance between two binary bitstrings u and v is denoted by $\| u \oplus v \|$.

The minimum distance of a convolution code is defined without reference to Hamming distance or Hamming weight. The use of Hamming distance or Hamming weight as an example here does not indicate any intent to limit an embodiment to these metrics as the only appropriate

metrics of the minimum distance of an error-correcting code. Another metric for a set of sequences whose elements are nonbinary, for example, would be the L_∞ norm, a measure of the maximum difference between elements. The L_∞ difference between the sequence $u = \{3, 4, 5\}$ and the sequence $v = \{10, 5, 1\}$ would be 7.

A decoding function f has a correction threshold of size t if it can correct any set of up to t errors. In other words, the decoding function f can successfully decode any corrupted codeword i whose errors are less than or equal to the correction threshold t of the decoding function. The error in a corrupted codeword i can be described as the offset δ from the nearest codeword c. In a binary block code where the Hamming weight of the corresponding offset δ is less than or equal to the bit correction threshold t, the decoding function f will successfully decode a corrupted codeword i to a codeword c in the set of codewords C. This concept is expressed mathematically as follows: given $c \in C$ and $\delta \in \{0, 1\}^n$ with $\| \delta \| \leq t$, then $f(c + \delta) = c$.

Generally, the Hamming distance between any two codewords in the set of codewords C is greater than two times the correction threshold (2t). If the Hamming distance between codewords were not greater than 2t, then a corrupted codeword i would exist that could be decoded into more than one codeword. The neighborhood of a codeword c comprises the subset of all possible corrupted codewords that the decoding function f maps to the codeword c. The neighborhood of a codeword c is denoted as $f^{-1}(c)$. The decoding function f is generally such that any corrupted codeword i in $f^{-1}(c)$ is closer to the codeword c than to any other codeword.

For example, given a message m that is one bit long ($k = 1$), a codeword c that is three bits long ($n = 3$), a set of two codewords C consisting of 000 and 111 ($C = \{000, 111\}$), and a decoding function f that computes majority, the correction threshold t for the decoding function t equals one bit error ($t = 1$). The decoding function f maps a corrupted codeword i consisting of three binary bits to 000 if at least two bits are 0 and to 111 if at least two bits are 1. The correction threshold t indicates that the decoding function f can correct a single bit error because changing a single digit in either 000 or 111 does not change the majority.

The coding efficiency of an error-correcting code is the ratio of the bit length of a message m to the bit length of a codeword c. The coding efficiency (k / n) measures the degree of redundancy in the error-correcting code. The lower the coding efficiency, the more redundancy in the codewords. The error-correcting code described in this example has a coding efficiency of 1/3. In general, codes that can correct a large number of errors have a low coding efficiency.

Error-correcting codes may be defined for non-binary spaces as well, and it is intended that the principles described here can be extended to such spaces.

It should be noted, however, that an error-correcting code traditionally involves changing a message m to a codeword c before transmission 30. In some situations, however, the translation function g cannot be applied effectively. For instance, when the message m itself contains errors, generating redundancy is problematic. The errors in the message m may well be propagated and reinforced by the redundancy in the corresponding codeword c. This situation exists in the case of biometric identification. Biometric readings are prone to errors and are typically not repeatable; accordingly, a biometric reading, also known as a template, should be considered a message m that includes errors. Thus, embodiments of the present invention do not use error-correcting codes in the traditional way.

Embodiments of the present invention use the decoding function f of an error-correcting code to relate an input pattern p to a codeword c. In some embodiments, the input pattern p is treated as a corrupted codeword i in an error-correcting code. In such embodiments, the decoding function f decodes the input pattern p into a codeword c as if the input pattern were a corrupted codeword i. In other embodiments, the input pattern p is mapped to a codeword c within the set of codewords C associated with a decoding function f. In these embodiments, such a codeword c is randomly selected from the set of codewords associated with a decoding function f. Embodiments of the invention do not make use of the translation function g or the reverse translation function g^{-1} of the error-correcting code. In consequence, such embodiments do not map a message m from the message space 10 to a codeword c from the set of codewords C in codeword space 20. Nor do such embodiments map a codeword c from the set of codewords C in codeword space 20 back to a message m from the message space 10. In fact, such embodiments do not use the message space 10 at all.

The commonest class of error-correcting codes are linear error-correcting codes. Almost all of the error-correcting codes presently used in practice are linear. It is convenient, although not necessary, to choose the decoding function of a linear error-correcting code for use in embodiments of the present invention. One property of linear error-correcting codes that is useful in a number of applications is that it is easy to select a codeword c uniformly at random from the set of codewords C.

Referring to FIG. 2, a fuzzy commitment scheme F includes mapping an input pattern p to a codeword c. The input pattern p may be any sort of input pattern, including a biometric

reading, a digital image, a signature drawn on a graphical input device, and the like. The n-bit string that represents the input pattern is referred to as a witness x. The witness x is mapped to a codeword c which is also an n-bit string; this mapping is referred to as the commitment portion of a fuzzy commitment scheme F. The codeword c can be considered to be the committed value of the witness x that maps to it.

A witness x can be uniquely expressed as the codeword c to which it maps. The offset δ is the offset between the witness x and that codeword c. The offset δ is an n-bit string that expresses the differences between the two n-bit strings that are the witness x and the codeword c. The witness x is likewise equivalent to the codeword c and the associated offset δ, mathematically expressed as $x = c + \delta$. The offset δ may be denoted mathematically as $\delta \in \{0, 1\}^n$ such that $x = c + \delta$.

An example geometric analogy for the mapping between a witness x and a codeword c according to one embodiment of the present invention is shown in FIG. 2. The set of codewords C are shown as the set of points c_1, c_2, c_3, and c_4 on the *u-v* plane; mathematically expressed as $C = \{c_1, c_2, c_3, c_4\}$. The witness x is shown as a point on the *u-v* plane in FIG. 2 with the coordinates (30, 595). The decoding function f associated with this example, but not shown, maps an input, which would traditionally be a corrupted codeword i, to the nearest codeword c within the set of codewords C. Accordingly, since in this example the input to the decoding function is the witness x, the decoding function f maps the witness x to the nearest codeword, codeword c_3. This process is mathematically expressed as $f(x) = c_3$. The codeword that the input maps to, in this case codeword c_3, is the codeword that is used to form the commitment of the input which is referred to in this application as the committed codeword. The offset δ between the witness x and the committed codeword c_3 is defined as $(u - 170, v + 95)$ where *u* and *v* are the two axes of the *u-v* plane. In one embodiment of a fuzzy commitment scheme F, the codeword c is concealed while the offset δ is left in plaintext, in other words, the offset δ is not encrypted.

Note that FIG. 2 illustrates a geometric analogy of a particularly simple case in which the decoding function f is always successful, having no minimum distance, in contravention to the usual case.

The codeword c, derived from the mapping of the witness x, is hashed with a one-way function known as a hash function h. A hash function h is a function that takes an input and produces an output such that is impractical to figure out what input corresponds to a given output

and to find another input that produces the same output. Known hash functions take an arbitrary length input and produce an output of fixed length. This process can be expressed mathematically as h: $\{0, 1\}^n \rightarrow \{0, 1\}^l$. Common classes of hash functions include hash functions based on block ciphers and hash functions with dedicated designs. Popular hash functions include SHA-1, MD5, RIPE-MD, HAVAL, and SNERFU. For a fuzzy commitment scheme F, hashing can be done with any appropriate hash function h. The output of a hash function h is known as the hash of the input. The hash of the codeword is referred to as h(c). In one embodiment, a fuzzy commitment scheme F accepts a witness x and a committed codeword c as input and produces a hash of the committed codeword h(c) and the offset δ between the witness x and the committed codeword c as output. Such a fuzzy commitment is expressed mathematically as $F(c, x) = (h(c), (x - c))$ where $F: (\{0, 1\}^n, \{0, 1\}^n) \rightarrow (\{0, 1\}^l, \{0, 1\}^n)$.

In some embodiments, the hash of the committed codeword h(c) in association with the offset δ is referred to as the blob y. In other embodiments, the blob y refers the hash of the committed codeword h(c) without an associated offset δ. The blob y may be used in a variety of applications. The original creation of a blob y through the application of a fuzzy commitment scheme F to a witness x is known as the commitment to a codeword c. The offset δ, that is part of the blob y in some embodiments, provides some information about the witness x. However, the blob y provides the remaining information needed to specify the witness x, namely the codeword c, in a concealed form only.

In one embodiment, a three step process is used to decommit a codeword c given a blob y and a second witness x′. In this embodiment, the blob y comprises the hash of the committed codeword c in association with the offset δ. First, the second witness x′ is translated by the offset δ into what we will call the corrupted codeword i. Second, the corrupted codeword i is decoded into a codeword c by the translation function f. We call the result of the translation of a second witness x′ by the offset δ a corrupted codeword i due to its relationship with the decoding function f. Where the corrupted codeword i is close enough to the original witness x, the decoding step will recover the original committed codeword c. Third, the codeword c is hashed to form a hashed codeword h(c). If the hash of the codeword h(c) matches the hash of the codeword h(c) in the blob y, then the decommitment of the blob y is successful. Otherwise, the decommitment fails. If f is an efficient decoding function, the decommitment will also be an efficient process.

Still referring to FIG. 2 as an example of one embodiment of the commitment process, the witness x is decoded into the committed codeword c_3. The offset δ between the witness x and the committed codeword c_3 is calculated. The committed codeword c_3 is hashed. A blob y (not shown) is created by associating the offset δ with the hash of the committed codeword $h(c_3)$.

Decommitment requires a blob y and use of the associated decoding function f. The blob y from FIG. 2 is a hash of the committed codeword $h(c_3)$ and the offset δ, mathematically represented as $y=(h(c_3), \delta)$. The offset δ reveals the location of the witness x relative to the committed codeword c_3, but does not reveal any information about the absolute location of the committed codeword c_3 or the witness x on the *u-v* plane. Thus, assuming that the hash function h is a secure one-way function, the only information that the blob y effectively reveals about the witness x is that it takes the form $(u + 170, v - 95)$ for some points (u, v). Subject to this constraint, the witness x could otherwise lie anywhere in plane.

Decommitment of the blob y, according to this embodiment, begins with the presentation of a second witness x′ that is likely to be near the unknown witness x. In FIG. 2, the second witness x′ is shown as a point on the *u-v* plane with the coordinates (40, 550). The second witness x′ is translated by the offset δ, just as the witness x was translated to reach the committed codeword c_3. The corrupted codeword i is represented as a point on the *u-v* plane with the coordinates (210, 455). Mathematically, $i = x' - \delta$. The decoding function f then decodes the corrupted codeword i into the nearest codeword, which in the geometric analogy of FIG. 2 is codeword c_3. Finally, a hash of the nearest codeword is compared to the hash of the codeword $h(c_3)$ in the blob y. When the second witness x′ is near the original unknown witness x, the nearest codeword will be the committed codeword c_3, the hashes will match, and the decommitment will be successful. Thus, the selection of a second witness x′ close to the witness x and the use of the decoding function f associated with the blob y in FIG. 2 enable c_3 to be decommitted.

In a simple embodiment in which the decoding function f maps any witness x to the nearest codeword c without limit on its distance from that nearest codeword c, the use of the offset δ may not be necessary. The use of this embodiment may be appropriate when a decoding function f that uses unconstrained codewords is selected. For example, in the example of FIG. 2, the second witness x′ is nearer the codeword c_3 than any other codeword. Thus, if the codewords in FIG. 2 are unconstrained, the corrupted codeword i would map directly to the codeword c_3, even without translation by the offset δ.

Referring to FIG. 3, in a more complex embodiment where the decoding function f maps a witness x to the nearest codeword c provided that its distance from the nearest codeword c falls within the minimum distance of the error-correcting code, the use of the offset δ may be useful. In this embodiment, a decoding function f that uses constrained codewords is selected. The dotted line circles surrounding each of the codewords C $\{c_1, c_2, c_3, c_4\}$ represent the boundaries of the area that maps to the included codeword c. Any point outside the dotted line circle surrounding a codeword c will not map to that codeword c, even if the point outside the dotted line circle is closer to the enclosed codeword c than to any other codeword. For example, the witness x in the figure does not fall within the boundaries of an area that will map to any codeword c. The decoding function f may output ϕ. Alternately, x may be mapped to a codeword c selected at random from the set of codewords C associated with the given decoding function f. If witness x is randomly mapped to codeword c_2, then the offset δ between x and c_2 can be represented as $(u + 470, v - 395)$.

Referring to FIG. 4, in the embodiment representing the decommitment process that corresponds to the commitment process explained by FIG. 3, a second witness x′ is translated by the offset δ in the decommitment process. When the second witness x′ is close to original point x, the second witness x′ can be reliably mapped to c_2 given the offset δ calculated in the commitment process. If the corrupted codeword i falls within the decoding constraints of the nearest codeword, the decoding function f decodes the corrupted codeword i into the nearest codeword. Here, the corrupted codeword i falls with the decoding constraints of codeword c_2, and so the decoding function f will map i to codeword c_2.

For a given witness x, such as an n-bit string representing a fingerprint template of a user, an attacker with knowledge of blob y = (h(c), δ) alone would be unable to find a second witness x′ to decommit c. On the other hand, if the user were to present her finger to a reading device that generates a second witness x′, in this case another n-bit string representing a fingerprint template of the user, it would be possible to decommit codeword c from blob y. Clearly, knowledge of blob y makes it possible to verify that a second witness x′ is close to the original witness x, and thus to authenticate the user. Speaking generally, the second witness x′ may be viewed as a fuzzy representation of the original witness x.

Referring to FIGS. 5A and 5B, the security of a commitment scheme depends on the commitment scheme being both concealing and binding. The property of concealment in a fuzzy commitment scheme F can be characterized as follows. Given that an attacker is able to

determine the codeword c from a fuzzy commitment scheme F whereby the codeword c is selected at random from the set of codewords C and the witness x represents a random binary n-bit string, mathematically $c \in_R C$ and $x \in_R \{0, 1\}^n$, in time T with probability $\rho(T)$. It is then possible for the attacker to invert h(c) from a random codeword z selected at random from the set of codewords C, mathematically $z \in_R C$, in time T with probability $\rho(T)$.

The time T and probability $\rho(T)$ required to invert h(c) are evident from the following. Since the witness x and the codeword c are selected independently and uniformly at random, it is clear that the offset δ, mathematically defined as $\delta = x' - c$, reveals no information about the codeword c. Therefore, the task of an attacker in determining the codeword c is equivalent to the task, given knowledge only of the hash of the codeword h(c), of finding a random codeword z selected from the set of codewords C such that the hash of the random codeword z equals the hash of the codeword c, mathematically h(z) = h(c). The underlying assumption in our derivation of the time T and probability $\rho(T)$ required to invert h(c), that it is hard to invert a hash of a codeword h(c) from the set of codewords C, is somewhat non-standard. It is, however, consistent with common security assumptions about hash functions, such as those provided by the random oracle model. The same result is reached using more canonical assumptions.

The amount of information about the witness x that is contained in a the committed codeword c determines the level of concealment in a fuzzy commitment scheme F. The amount of information about the witness x contained in the committed codeword c depends on the number of codewords or, stated in another way, the size of the set of codewords C. The larger the set of codewords, the more information that the committed codeword contains about the witness x.

A comparison of FIG. 5A and FIG. 5B illustrates this concept through another geometric analogy. FIG. 5A shows a witness x and a set of one codeword $C = \{c\}$, both as points on a plane. A decoding function f will map the witness x in FIG. 5A to the nearest codeword c. Since there is a single codeword c in the set of codewords C in FIG. 5A, x must be mapped to codeword c and we have no information about the true location of the witness x on the plane. In comparison, FIG. 5B shows a witness x and a set of four codewords, all as points on a plane. A decoding function f will map the witness x in FIG. 5B to the nearest codeword c_1. Since there are four codewords in the set of codewords $C = \{c_1, c_2, c_3, c_4\}$ in FIG. 5B and the witness x is mapped to codeword c_1, the location of the witness x is bounded. Clearly, the codeword c_1 contains a lot more information about the witness x in FIG. 5B than codeword c does in FIG. 5A

even though the witness x and the codeword it maps to have the same relationship in FIG. 5A and FIG. 5B.

Since the hash of the committed codeword h(c) is always used and the unhashed committed codeword c is never used, the information about the witness x in the committed codeword c represents concealed information. That is, the information about the witness x in the committed codeword c is concealed by the hash. Thus, the codeword c represents information about the witness x that is concealed. Simply stated, the larger the set of codewords C, the more information about the witness x that is concealed.

Since error-correcting codes require that a binary set of codewords C contain 2^k codewords, k describes the size of a set of codewords C. Thus, a higher k represents more codewords and more concealed information about the witness x. Effectively, k is the parameter that dictates the level of concealment in a fuzzy commitment scheme F. For most applications, a k value of eighty should provide an adequate level of security. Under common assumptions about hash functions, such as the random oracle model, this security level will require an attacker seeking to match the committed codeword c in the blob y an average of 2^{79} hash function h computations. This number of calculations is comparable to the computational effort required to factor RSA-1024 or find a collision in SHA-1.

Note that in some embodiments, where it is not necessary to protect the codeword c itself, the codeword c must still be drawn from a large set of codewords C in order to conceal the witness x. Consider, for example, a straightforward fingerprint authentication scenario meant to model the use of hashed passwords on UNIX systems. Here, the blob y includes an offset δ and is stored on a server. In order to demonstrate her identity, a user must simply present the server with a fingerprint image that successfully decommits the codeword c. The codeword c must be drawn from a large enough set of codewords C to ensure that the blob y does not reveal the witness x, which in this case is the fingerprint image itself. If the set of codewords C, as described by k, is small, then an attacker can guess the codeword c and extract the witness x from the blob y.

A commitment scheme is conventionally defined as binding if it is infeasible for any polynomially bounded player to produce valid decommitments of the commitment for two distinct witnesses x_1 and x_2. A fuzzy commitment scheme F applies a stronger notion of binding. A fuzzy commitment scheme F is defined as strongly binding if it is infeasible for any polynomially bounded player to produce a witness collision. A witness collision is a pair of

witnesses x_1 and x_2 that are not close but that nonetheless both produce the same hash of a codeword h(c). A pair of witnesses x_1 and x_2 are close if the decoding function f produces the same codeword c from each of the translated witnesses, mathematically denoted as $f(x_1 - \delta) = f(x_2 - \delta)$. In other words, closeness is defined as within the maximum distance allowed by the underlying error-correcting code. This definition of strongly binding subsumes the conventional definition of binding. Strong binding may, of course, also be defined in a conventional commitment scheme by allowing a witness collision to include any two witnesses, x_1 and x_2, that are distinct. Consequently, if a fuzzy commitment scheme F is strongly binding, then a fuzzy commitment scheme F is also binding.

Further, a fuzzy commitment scheme F is strongly binding if the associated hash function h is collision resistant. If an attacker is capable of finding a witness collision, then the attacker can find a collision on the hash function h. The length *l* of the binary bit string created by the hash function h dictates how hard it is to find a witness collision. Effectively, *l* is the parameter that dictates the strength of the binding in a fuzzy commitment scheme F. Under the common assumption that the most effective means of finding a collision in a hash function is a birthday attack whereby pairs of hashes are compared in an effort to find a match, $2^{l/2}$ hashes, or calculations, are required to find a match. Hence, a *l* value of one hundred sixty, which corresponds to the image length of SHA-1, results in a minimum of about 2^{80} calculations to match a hash. A strong binding commitment scheme is particularly useful for biometric authentication scenarios.

In the context of an error-correcting code, resilience refers the maximum level of corruption, or number of errors, in a corrupted codeword i with which the decoding function f can reconstruct the codeword c. This is also known as the error correction threshold t of the error-correcting code. The error correction threshold t is bounded by the minimum distance between codewords in the set of codewords C (known as the minimum distance of the code). In the context of a fuzzy commitment scheme F, resilience refers to the maximum offset δ of a witness x from a codeword c with which the decoding function can derive the codeword c from the witness x. The resilience of a fuzzy commitment scheme is clearly bounded by the error correction threshold t of the error-correcting code used in its construction.

Again, since error-correcting codes require that a binary set of codewords C must contain 2^k codewords, k describes the size of a set of codewords C. Thus, a lower k represents fewer codewords and potentially a greater minimum distance of the code which represents a greater

potential error-correction threshold t and a greater potential allowable offset δ. A lower k also represents a lower level of security in a fuzzy commitment scheme F. Clearly, the resilience of a fuzzy commitment scheme F is inversely related to its level of concealment. A fuzzy commitment scheme F achieves a tradeoff between resilience and concealment by varying k.

In general, the larger the coding efficiency k/n, the larger the minimum distance achievable in an error-correcting code. This is logical since coding efficiency k/n is proportional to the redundancy permitted in the code. The value n of an error-correcting code is typically fixed by the particular application. Similarly, k should be approximately 80 to prevent brute-force inversion attacks against the underlying hash function h in a fuzzy commitment scheme. Where the parameters k and n are fixed, there is no straightforward way to determine the most efficient error-correcting code. The design of codes to handle particular parameter sets is a broad research topic covered in some degree by classic texts. In general, practitioners resort to tables of the best known codes.

To get a sense of the level of resilience attainable in a practical setting, consider an application with a n value of 540. The n value of 540 roughly corresponds to a lower bound on the amount of information in a typical template extracted by the latest generation of fingerprint scanning chips manufactured by Veridicom. A practitioner may use a table of BCH codes, an efficiently computable class of error-correcting codes, and discover an error-correcting code with a k value of 76, a n value of 511, and a correction threshold t of 85 bits. The value of k in the selected error-correcting code offers an acceptable security level for a fuzzy commitment scheme F. A set of codewords C with a length of 511 bits may be used if some data from the application is truncated or compressed. Thus, the selected BCH error-correcting code would enable a practitioner to construct a fuzzy commitment scheme F that tolerates errors in any witness x of up to almost 17% of the component bits.

Here, each witness x has been selected uniformly at random from the set of n-bit binary strings. If a witness x were instead drawn from some non-uniform distribution D within the set of n-bit binary strings, then the security level of a fuzzy commitment scheme F will be affected to some degree. Some distributions will not result in a significant diminution in the security parameter k, while others will yield a lesser security level. A good security analysis will, in general, require detailed knowledge of the distribution of witnesses in the relevant application. Nonetheless, if a non-uniform distribution D is only slightly non-uniform, only a slight diminution in security will result. Larger dimunitions in security can be compensated for by

increasing k. Of course, increasing k may reducing the resilience of the fuzzy commitment scheme F.

Similarly, the differences between the original witness x and a subsequent witness x′ have been assumed to be random here. Note, however, that when the differences between the original witness x and a subsequent witness x′ can be correlated, it is sometimes possible to construct a fuzzy commitment scheme F that achieve a higher level of resilience than the error correction threshold t of the selected error-correcting code. This is possible because correlations in the differences restrict the number of likely error patterns. If errors tend to occur in sequence, for example, then it is advantageous to use Reed-Solomon codes. Reed-Solomon codes are well-known for their use in the digital recording media such as compact discs, where so-called burst errors are common. An advantage of Reed-Solomon codes is that much progress has been made recently in achieving probable error correction beyond the error correction threshold t for this class of code. In certain cases, it may even be possible to use such codes to achieve good error correction under independence of bits in e.

Referring to FIG. 6, one aspect of a fuzzy commitment includes receiving an input (STEP 40). In one embodiment, the input is the witness x. In one embodiment, the witness x is a first biometric reading, such as an iris scan, a measurement of certain features of a fingerprint, or the digital scan of an image such as a signature. In another embodiment, the witness x is a signature captured by a graphical interface. In another embodiment, the witness x is a digital image or a profile of mutable executable code such as a digital virus profile.

The input is mapped to a codeword (STEP 50). The mapping consists of randomly selecting a codeword c from the set of codewords C associated with an error-correcting code. In one embodiment, the set of codewords is constrained. In another embodiment, the decoding function f is part of a linear error-correcting code. In another embodiment, the decoding function f is from an error-correcting code with isometric codeword neighborhoods.

An offset δ between the input and the codeword c to which it was mapped is calculated (STEP 60). In one embodiment, the input, the codeword c, and the offset δ are all represented as binary n-bit strings. In another embodiment, the codeword c and the offset δ are all represented as binary n-bit strings and the input is represented as a longer binary string. In one variation of the foregoing embodiment, the binary string representing the input is truncated prior to the calculation of the offset δ. In another variation of the foregoing embodiment, the binary string representing the input is compressed prior to the calculation of the offset δ.

The codeword c is hashed (STEP 70). Hashing can be performed using any appropriate hash function h. In one embodiment, the hash function produces a binary string with a length l of approximately 160 bits.

Referring to FIG. 7, another aspect of a fuzzy commitment includes receiving an input (STEP 40). A codeword is derived from the input (STEP 90). In one embodiment, the derivation is performed by a decoding function f of an error-correcting code. In one embodiment, the set of codewords associated with the error-correcting code is constrained. In another embodiment, the decoding function f is part of a linear error-correcting code. In another embodiment, the decoding function f is from an error-correcting code with isometric codeword neighborhoods.

In one embodiment, the error-correcting code has a dimension d in which codewords are of the form <Ra_1, Ra_2, ..., Ra_d> such that a_i is an integer and R is a real-valued code parameter and the decoding function f as applied to vector <x_1, x_2, ..., x_d> simply rounds each element x_i to the integer a_iR that is closest. In a related embodiment, where there are ambiguities, a deterministic or randomized tie-breaking algorithm is used, or both possibilities are checked.

In one embodiment, the decoding function f receives the input directly for decoding. In one variation of the foregoing embodiment, an offset δ between the codeword c and the input from which it was derived is calculated (STEP 60). In a further variation of the foregoing embodiment, the offset δ is stored. In another embodiment, the input is translated by a known offset δ (STEP 110) and a decoding function f of an error-correcting code derives the codeword from the translated input. In the foregoing embodiment, the translated input is a corrupted codeword i. In one variation of the foregoing embodiment, the known offset δ is retrieved from storage for use. The codeword c is hashed (STEP 70).

Referring to FIG. 8, an embodiment of a method for registering an input for later authentication includes receiving an input (STEP 40). The input is mapped to a codeword (STEP 50), and an offset δ between the input and the codeword c to which it was mapped is calculated (STEP 60). The codeword c is hashed (STEP 70).

The offset δ and the hash of the codeword h(c) are stored (STEP 80). In one embodiment, the hash of the codeword h(c) is stored on a hard disk. In one embodiment, the offset δ and the hash of the codeword h(c) are stored together. In another embodiment, the offset δ and the hash of the codeword h(c) are stored separately. In one embodiment, the hash of the

codeword h(c), the offset, or both are stored on a CD-ROM. In another embodiment, In one embodiment, the hash of the codeword h(c), the offset, or both are stored on a network. In another embodiment, In one embodiment, the hash of the codeword h(c), the offset, or both are stored on a smartcard. In another embodiment, In one embodiment, the hash of the codeword h(c), the offset, or both are stored on a personal digital assistant. In another embodiment, In one embodiment, the hash of the codeword h(c), the offset, or both are stored on a magnetic strip, such as might be attached to a credit card-sized card. In another embodiment, In one embodiment, the hash of the codeword h(c), the offset, or both are stored on a bar code.

Referring to FIG. 9, an embodiment of a method for registering an input for later authentication includes receiving an input (STEP 40). A codeword is derived from the input as described above (STEP 90). The codeword c is hashed (STEP 70), and the hash of the codeword h(c) is stored (STEP 100).

Referring to FIG. 10, an embodiment of a method for authenticating a registered input includes receiving an input as described above (STEP 40). In some embodiments, since an input has already been registered, the input is a second witness x′.

A codeword is derived from the input (STEP 90). In one embodiment, a decoding function f receives the input directly for decoding. In another embodiment, the input is translated by an offset δ (STEP 110) and then the input, now a corrupted codeword i, is communicated to a decoding function f for decoding.

The codeword c is hashed as described above (STEP 70). The hash of the codeword h(c) is compared to a stored hash (STEP 120). In one embodiment, the two hashes are considered a match where the entirety of each hash is a duplicate of the other. In one embodiment, the two hashes are considered a match where a portion of each hash is a duplicate of the other.

If the hash of the codeword h(c) matches a stored hash, the input is authenticated (STEP 130). In one embodiment, authentication results in a signal being returned in response to the input. In another embodiment, authentication results in permissions being granted in response to the input.

Referring to FIG. 11, an embodiment of a method for registering an input includes receiving an input as described above (STEP 40). In some embodiments, since an input is being registered, the input is a witness x. The input is mapped to a codeword as described above (STEP 50). An offset δ between the input and the codeword c to which it was mapped is calculated (STEP 60). The offset δ between the input and the codeword is stored (STEP 140). In

one embodiment, the stored offset δ is appended to the public key. The codeword c is hashed (STEP 70).

A key is generated from the hash of the codeword h(c) (STEP 150). In one embodiment, the hash of the codeword h(c) itself may serve as the symmetric encryption key. In another embodiment, the key pair is generated with a symmetric encryption algorithm and the hash of the codeword h(c). In another embodiment, the key pair is generated with an asymmetric encryption algorithm and the hash of the codeword h(c). In another embodiment, the key pair is generated by feeding the hash of the codeword h(c) as a seed to an asymmetric key generation algorithm. In one embodiment, the asymmetric key generation algorithm is an RSA key pair generation algorithm.

In one embodiment, the key that is generated is transmitted to an authentication entity (STEP 160). In a variation of the foregoing embodiment, the transmitted key is the public key generated with an asymmetric encryption algorithm.

Referring to FIG. 12, an embodiment of a method for registering an input includes receiving an input (STEP 40), and deriving a codeword from the input (STEP 90). The codeword c is hashed (STEP 70), and a key is generated from the hash of the codeword h(c) (STEP 150). In one embodiment, the key that is generated is transmitted to an authentication entity (STEP 160).

Referring to FIG. 13, an embodiment of a method for responding to a challenge message includes receiving a message m from an authentication entity (STEP 170). In one embodiment, the challenge message is encrypted. In another embodiment, the challenge message is in plaintext. Plaintext is unencrypted information, and is sometimes also referred to as cleartext. In one embodiment, the challenge message is time dependent. In another embodiment, the challenge message includes an encrypted message and an offset δ in plaintext. In another embodiment, the challenge message does not include an offset in plaintext.

An input is received (STEP 40), and a codeword is derived from the input (STEP 90). In one embodiment, the derivation is performed by a decoding function f of an error-correcting code. In one embodiment, the decoding function f receives the input directly for decoding. In another embodiment, the input is translated by an offset δ (STEP 110), and then the translated input, known as a corrupted codeword i, is communicated to the decoding function f for decoding. In one variation of the foregoing embodiment, the offset δ is taken from the challenge message.

The codeword c is hashed (STEP 70), and a key is generated from the hash of the codeword h(c) (STEP 150). A key is applied to the message m to create a response (STEP 180). In one embodiment, a symmetric key is applied to the message m to create a response. In another embodiment, a private key is applied to the message m to create a response. In a variation of the foregoing embodiments, the application of the key to the message m consists of the use of the key to encrypt the message m. In another variation of the foregoing embodiment, the application of the key to the message m consists of the use of the key to decrypt the message m. In another variation of the foregoing embodiment, the application of the key to the message m consists of the use of the key to sign the message m.

The response is transmitted (STEP 190). In one embodiment, the response is transmitted back to the authentication entity. In another embodiment, the response is transmitted to another entity. Transmission can be over a wired or wireless network or communications medium, for example over a packet-based network, or via a direct connection.

Referring to FIG. 14, an embodiment of a method for encrypting a message includes receiving an input as described above (STEP 40). In some embodiments, the input is a witness x. The input is mapped to a codeword (STEP 50), and an offset δ between the input and the codeword c to which it was mapped is calculated (STEP 60).

The codeword c is hashed (STEP 70), and a message is encrypted (STEP 200). In one embodiment, the message is encrypted using a symmetric encryption algorithm and the hash of the codeword h(c). In a variation of the foregoing embodiment, the message is encrypted using a symmetric encryption algorithm, the hash of the codeword h(c), and the offset δ between the input and the codeword c. In another embodiment, the message is encrypted using an asymmetric encryption algorithm and the hash of the codeword h(c). In a variation of the foregoing embodiment, the message is encrypted using an asymmetric encryption algorithm, the hash of the codeword h(c), and the offset δ between the input and the codeword c. In one embodiment, the offset δ is included in plaintext as a portion of the encrypted message. In another embodiment, the offset δ is stored but not included in the encrypted message.

Referring to FIG. 15, an embodiment of a method for encrypting a message includes receiving an input as described above (STEP 40). In some embodiments, the input is a witness x.

A codeword is derived from the input as described above (STEP 90). In one embodiment, the derivation may be performed by a decoding function f of an error-correcting code. In one embodiment, the decoding function f receives an input directly for decoding. In

another embodiment, the input is translated by an offset δ (STEP 110) and then the translated input, known as a corrupted codeword i, is communicated to the decoding function f for decoding. The codeword c is hashed (STEP 70), and a message is encrypted (STEP 200).

Referring to FIG. 16, an embodiment of a method for decrypting an encrypted message includes receiving an encrypted message (STEP 210). In one embodiment, the encrypted message includes an offset δ in plaintext. In another embodiment, the encrypted message does not include an offset δ in plaintext. An input is received (STEP 40).

A codeword is derived from the input pattern as described above (STEP 90). In one embodiment, the input is communicated directly to decoding function f for decoding. In another embodiment, the input is translated by an offset δ (STEP 110) and then the translated input, known as a corrupted codeword i, is communicated to the decoding function f for decoding. In one variation of the foregoing embodiment, the offset δ is taken directly from a plaintext portion of the encrypted message. In another variation of the foregoing embodiment, the offset δ is taken from storage and is not included as a portion of the encrypted message.

The codeword c is hashed (STEP 70), and key pair is generated from the hash of the codeword h(c). The encrypted message is decrypted (STEP 220). In one embodiment, a private key is used to decrypt the message m. In another embodiment, a private key is used to sign the message m. In another embodiment, a public key is used to decrypt the message m.

Referring to FIG. 17, an embodiment of a method for deriving a key includes receiving an input as described above (STEP 40). The input is mapped to a codeword (STEP 50), and an offset δ between the input and the codeword c to which it was mapped is calculated as described above. In one embodiment, the offset δ is stored for later use. The codeword c is hashed (STEP 70).

A key is generated as described above (STEP 150). In one embodiment, the message is encrypted with the key. In one variation of the foregoing embodiment, the message is later decrypted with the key. In another variation of the foregoing embodiment, the message is later decrypted with a related key. In another embodiment, the message is decrypted with the key. In another embodiment, the message is signed with the key.

Referring to FIG. 18, an embodiment of a method for deriving a key includes receiving an input as described above (STEP 40). A codeword is derived from the input pattern as described above (STEP 90). In one embodiment, the input is communicated directly to decoding function f for decoding. In another embodiment, the input is translated by an offset δ (STEP 110) and then

the translated input, known as a corrupted codeword i, is communicated to the decoding function f for decoding. In one embodiment, an offset δ between the input and the codeword c to which it was mapped is calculated as described above. In one embodiment, the offset δ is stored for later use. The codeword c is hashed (STEP 70), and a key is generated (STEP 150).

Apparatus that embody the foregoing methods and variations thereto are within the scope of the invention.

Referring to FIG. 19 for example, an embodiment of an apparatus for registering an input includes an input device (230) for receiving an input from a user. In one embodiment, the input device is a biometric scanning device. The input corresponding to such an embodiment may include an iris scan or a measurement of certain features of a fingerprint. In another embodiment, the input device is a non-biometric scanning device. The input corresponding to such an embodiment may include a digital scan of an image such as a signature. In another embodiment, the input device is a graphical input device. Variations according to this embodiment may include a touch sensitive screen and a heat sensitive screen. The input corresponding to this embodiment may be a signature captured by a graphical interface.

The apparatus also includes a mapper (240) in signal communication with the input device. The mapper maps the input pattern to a first codeword from the set of codewords C associated with an error-correcting code. In one embodiment, the mapper uniformly at random selects a codeword c. In another embodiment, the mapper selects among a set of codewords C that is constrained. In another embodiment, the mapper selects among a set of codewords C that is part of a linear error-correcting code. In another embodiment, the mapper selects among a set of codewords C that have isometric codeword neighborhoods.

The apparatus may also include, in some embodiments, an offset calculator (250) in signal communication with the mapper. The offset calculator of some embodiments calculates the offset δ between the input and the codeword c to which it was mapped. In one embodiment, the offset calculator accepts two binary n-bit strings representing the input and the codeword c and produces a third binary n-bit strings that represents the offset δ. In another embodiment, the offset calculator accepts a binary n-bit strings representing the codeword c and longer bit string representing the input, the offset calculator then truncates the input string and produces a second binary n-bit strings that represents the offset δ. In another embodiment, the offset calculator accepts a binary n-bit strings representing the codeword c and longer bit string representing the

input, the offset calculator then compresses the input string and produces a second binary n-bit strings that represents the offset δ.

The apparatus also includes a hasher (260) in signal communication with the mapper. The hasher hashes the first codeword. The hasher may apply any appropriate one-way (hash) function h to the codeword c. In one embodiment, the hasher applies a hash function h built around a block cipher. In another embodiment, the hasher applies a hash function h which has a dedicated design. In one embodiment, the hasher may accept an n-bit string representing the codeword c and produce a longer l-bit string representing the hash of the codeword h(c). In one embodiment, the hasher produces a binary string with a length l of approximately 160 bits.

The apparatus also includes a storage device (270) in signal communication with the hasher. The storage device stores a hash of the first codeword. In one embodiment, the storage device may also store an offset between the input pattern and the first codeword. In such an embodiment, the storage device (270) must be in signal communication with the offset calculator (250). In another embodiment, the offset δ may be stored in a separate storage device (280) in signal communication with an offset calculator. In such an embodiment, the storage device (280) must be in signal communication with the offset calculator (250). In one embodiment the storage is temporary, only long enough to enable reliable transmission.

Variations, modifications, and other implementations of what is described herein will occur to those of ordinary skill in the art without departing from the spirit and the scope of the invention as claimed. Accordingly, the invention is to be defined not by the preceding illustrative description but instead by the spirit and scope of the following claims.

Claims

What is claimed is:

1. A method for generating a fuzzy commitment from an input pattern, the method comprising the steps of:

(a) receiving an input pattern from a user;

(b) mapping the input pattern to a first codeword, said first codeword selected at random from a plurality of codewords, said plurality of codewords being associated with an error-correcting code;

(c) calculating an offset between the input pattern and the first codeword; and

(d) hashing the first codeword to produce a hash of the first codeword.

2. The method of claim 1 wherein the step of receiving an input pattern comprises receiving an input pattern selected from the group of metrics associated with a user consisting of:

(i) a first measurement of a biometric,

(ii) a first measurement of a pattern of behavior,

(iii) a digital image, and

(iv) a profile of a mutable executable code.

3. The method of claim 1, the method further comprising the step of storing the offset.

4. The method of claim 3, the method further comprising the step of storing the offset and the hash of the first codeword.

5. The method of claim 3, the method further comprising the step of generating a key using an encryption algorithm and the hash of the first codeword as a key generation seed.

6. The method of claim 5, the method further comprising the step of encrypting a message using the key.

7. The method of claim 6, the method further comprising the step of decrypting the encrypted message using the key.

8. The method of claim 5, the method further comprising the step of decrypting an encrypted message using the key.

9. The method of claim 5, the method further comprising the step of signing a message using the key.

10. The method of claim 5 wherein the step of generating a key using an encryption algorithm and the hash of the first codeword as the seed comprises generating a key pair using an asymmetric encryption algorithm and the hash of the first codeword as a key generation seed,

said key pair comprising a public key and a private key.

11. The method of claim 10, the method further comprising the step of transmitting the public key from the key pair to an authentication entity.

12. The method of claim 1, the method further comprising the steps of:

(e) generating a key using an encryption algorithm and the hash of the first codeword as the seed;

(f) encrypting a message with the key; and

(g) including the offset in plaintext with the encrypted message.

13. A method for generating a fuzzy commitment from an input pattern, the method comprising the steps of:

(a) receiving an input pattern from a user;

(b) deriving a first error-correcting codeword by applying a decoding function to the input pattern; and

(c) hashing the first error-correcting codeword to produce a hash of the first codeword.

14. The method of claim 13 wherein the step of receiving an input pattern comprises receiving an input pattern selected from the group of metrics associated with a user consisting of:

(i) a first measurement of a biometric, and

(ii) a first measurement of a pattern of behavior,

(iii) a digital image, and

(iv) a profile of a mutable executable code.

15. The method of claim 13, the method further comprising the step of storing the hash of the first error-correcting codeword.

16. The method of claim 13, the method further comprising the steps of comparing the hash of the first error-correcting codeword to a stored hash and authenticating the input pattern when the hash of the first error-correcting codeword equals the stored hash.

17. The method of claim 13, the method further comprising, prior to step (b), the step of translating the input pattern by an offset; and wherein step (b) comprises deriving a first error-correcting codeword by applying a decoding function to the translated input pattern.

18. The method of claim 17, the method further comprising the steps of comparing the hash of the first error-correcting codeword to a stored hash and authenticating the input pattern when the hash of the first error-correcting codeword equals the stored hash.

19. The method of claim 17, the method further comprising the step of generating a key using an encryption algorithm and the hash of the first codeword as a key generation seed.

20. The method of claim 19, the method further comprising the step of encrypting a message using the key.

21. The method of claim 20 the method further comprising the step of decrypting the encrypted message using the key.

22. The method of claim 19 the method further comprising the step of decrypting an encrypted message using the key.

23. The method of claim 22 wherein the encrypted message includes the offset in plaintext.

24. The method of claim 19, the method further comprising the step of signing a message using the key.

25. The method of claim 19 wherein the step of generating a key using an encryption algorithm and the hash of the first codeword as a key generation seed comprises generating a key pair using an asymmetric encryption algorithm and the hash of the first codeword as a key generation seed, said key pair comprising a public key and a private key.

26. The method of claim 25, the method further comprising the step of transmitting the public key from the key pair to an authentication entity.

27. An apparatus for generating a fuzzy commitment from an input pattern, the apparatus comprising:

an input device, said input device receiving an input pattern from a user;

a mapper in signal communication with the input device, said mapper mapping the input pattern to a first codeword selected at random from the plurality of codewords associated with an error-correcting code;

an offset calculator in signal communication with the mapper, said offset calculator calculating an offset between the input pattern and the first codeword to which the mapper maps the input pattern; and

a hasher in signal communication with the mapper, said hasher producing a hash of the first codeword by applying a hash function to the first codeword.

28. The apparatus of claim 27 wherein the input device comprises a reader that measures an input pattern selected from the group of metrics associated with a user consisting of:

(i) a first measurement of a biometric, and

(ii) a first measurement of a pattern of behavior,

(iii) a digital image, and

(iv) a profile of a mutable executable code.

29. The apparatus of claim 27, the apparatus further comprising a storage device in signal communication with the offset calculator, said storage device storing the offset.

30. The apparatus of claim 27, the apparatus further comprising a storage device in signal communication with the offset calculator and the hasher, said storage device storing the offset and the hash of the first codeword.

31. The apparatus of claim 29, the apparatus further comprising a key generator in signal communication with the hasher, said key generator generating a key using an encryption algorithm and the hash of the first codeword as a key generation seed.

32. The apparatus of claim 31, the apparatus further comprising an encryption device in signal communication with the key generator, said encryption device encrypting a message using the key.

33. The apparatus of claim 32, the apparatus further comprising a decryption device in signal communication with the key generator, said decryption device decrypting the encrypted message using the key.

34. The apparatus of claim 31, the apparatus further comprising a decryption device in signal communication with the key generator, said decryption device decrypting an encrypted message using the key.

35. The apparatus of claim 31, the apparatus further comprising a signature device in signal communication with the key generator, said signature device signing a message using the key.

36. The apparatus of claim 31 wherein the key generator generates a key pair using an asymmetric encryption algorithm and the hash of the first codeword as a key generation seed, said key pair comprising a public key and a private key.

37. The apparatus of claim 36, the apparatus further comprising a transmission device in signal communication with the key generator, the transmission device transmitting the public key to an authentication entity.

38. The apparatus of claim 27, the apparatus further comprising:

a key generator in signal communication with the hasher, said key generator generating a key using an encryption algorithm and the hash of the first codeword as a key generation seed;

an encryption device in signal communication with the key generator, said encryption device encrypting a message using the key; and

a concatenator in signal communication with the encryption device and the offset calculator, said concatenator joining the encrypted message and the offset.

39. An apparatus for generating a fuzzy commitment from an input pattern, the apparatus comprising:

an input device, said input device receiving an input pattern from a user;

a codeword generator in signal communication with the input device, said codeword generator producing a first codeword by applying a decoding function of an error-correcting code to the input pattern; and

a hasher in signal communication with the codeword generator, said hasher producing a hash of the first codeword by applying a hash function to the first codeword.

40. The apparatus of claim 39 wherein the input device comprises a reader that measures an input pattern selected from the group of metrics associated with a user consisting of:

(i) a first measurement of a biometric, and

(ii) a first measurement of a pattern of behavior,

(iii) a digital image, and

(iv) a profile of a mutable executable code.

41. The apparatus of claim 39, the apparatus further comprising a storage device in signal communication with the hasher, said storage device storing the hash of the first codeword.

42. The apparatus of claim 39, the apparatus further comprising a comparator in signal communication with the hasher, said comparator comparing the hash of the first codeword to a stored hash and authenticating the input pattern when the hash of the first codeword matches the stored hash.

43. The apparatus of claim 39, the apparatus further comprising a translator in signal communication with the input device, said translator producing a translated input pattern by translating the input pattern by an offset, and wherein the codeword generator is in signal communication with the input device via the translator, said codeword generator producing a first codeword by applying a decoding function of an error-correcting code to the translated input pattern.

44. The apparatus of claim 43, the apparatus further comprising a comparator in signal communication with the hasher, said comparator comparing the hash of the first codeword to a stored hash and authenticating the input pattern when the hash of the first codeword matches the stored hash.

45. The apparatus of claim 43, the apparatus further comprising a key generator in signal communication with the hasher, said key generator generating a key using an encryption algorithm and the hash of the first codeword as the key generation seed.

46. The apparatus of claim 45, the apparatus further comprising an encryption device in signal communication with the key generator, said encryption device encrypting a message using the key.

47. The apparatus of claim 46, the apparatus further comprising a decryption device in signal communication with the key generator, said decryption device decrypting the encrypted message using the key.

48. The apparatus of claim 45, the apparatus further comprising a decryption device in signal communication with the key generator, said decryption device decrypting an encrypted message using the key.

49. The apparatus of claim 48 wherein the translator produces the translated input pattern by translating the input pattern by an offset, said offset included as a portion of the encrypted message.

50. The apparatus of claim 45, the apparatus further comprising a signature device in signal communication with the key generator, said signature device signing a message using the key.

51. The apparatus of claim 45 wherein the key generator generates a key pair using an asymmetric encryption algorithm and the hash of the first codeword as a key generation seed, said key pair comprising a public key and a private key.

52. The apparatus of claim 51, the apparatus further comprising a transmission device in signal communication with the key generator, the transmission device transmitting the public key to an authentication entity.

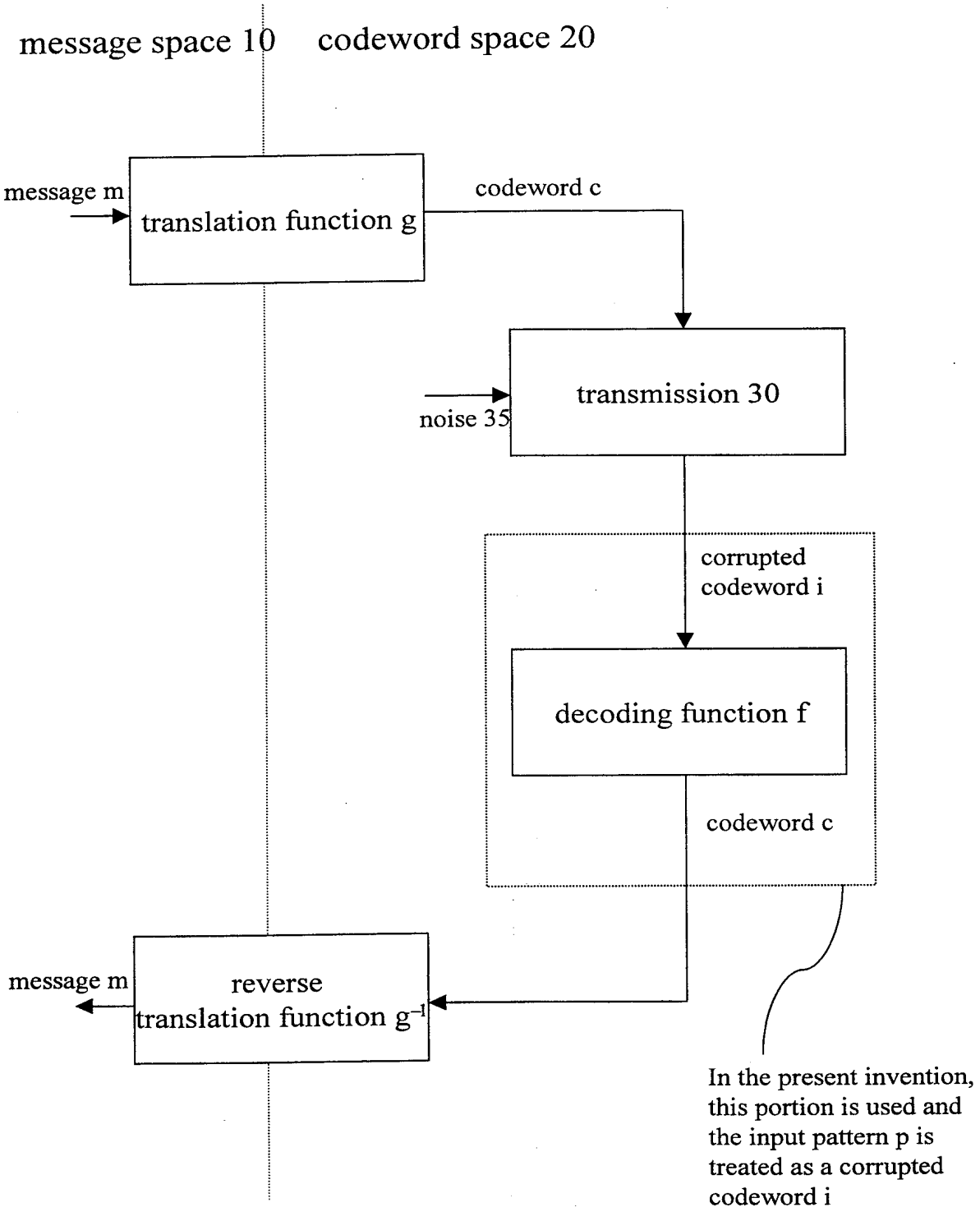

message space 10
codeword space 20
message m
translation function g
codeword c
transmission 30
noise 35
corrupted codeword i
decoding function f
codeword c
message m
reverse translation function g^{-1}
In the present invention, this portion is used and the input pattern p is treated as a corrupted codeword i

FIG. 1

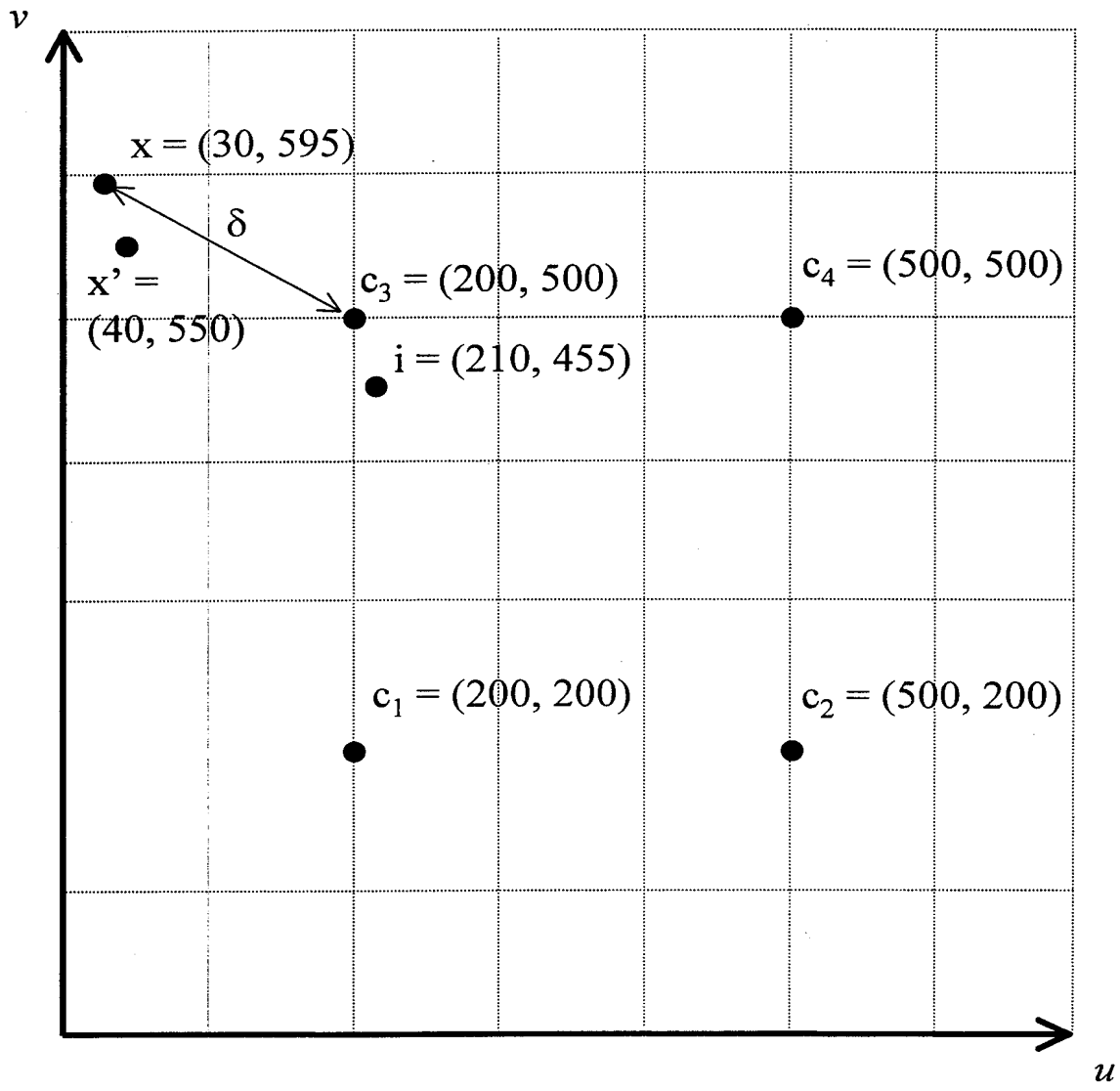

v
x = (30, 595)
δ
x' =
(40, 550)
c3 = (200, 500)
c4 = (500, 500)
i = (210, 455)
c1 = (200, 200)
c2 = (500, 200)
u

FIG. 2

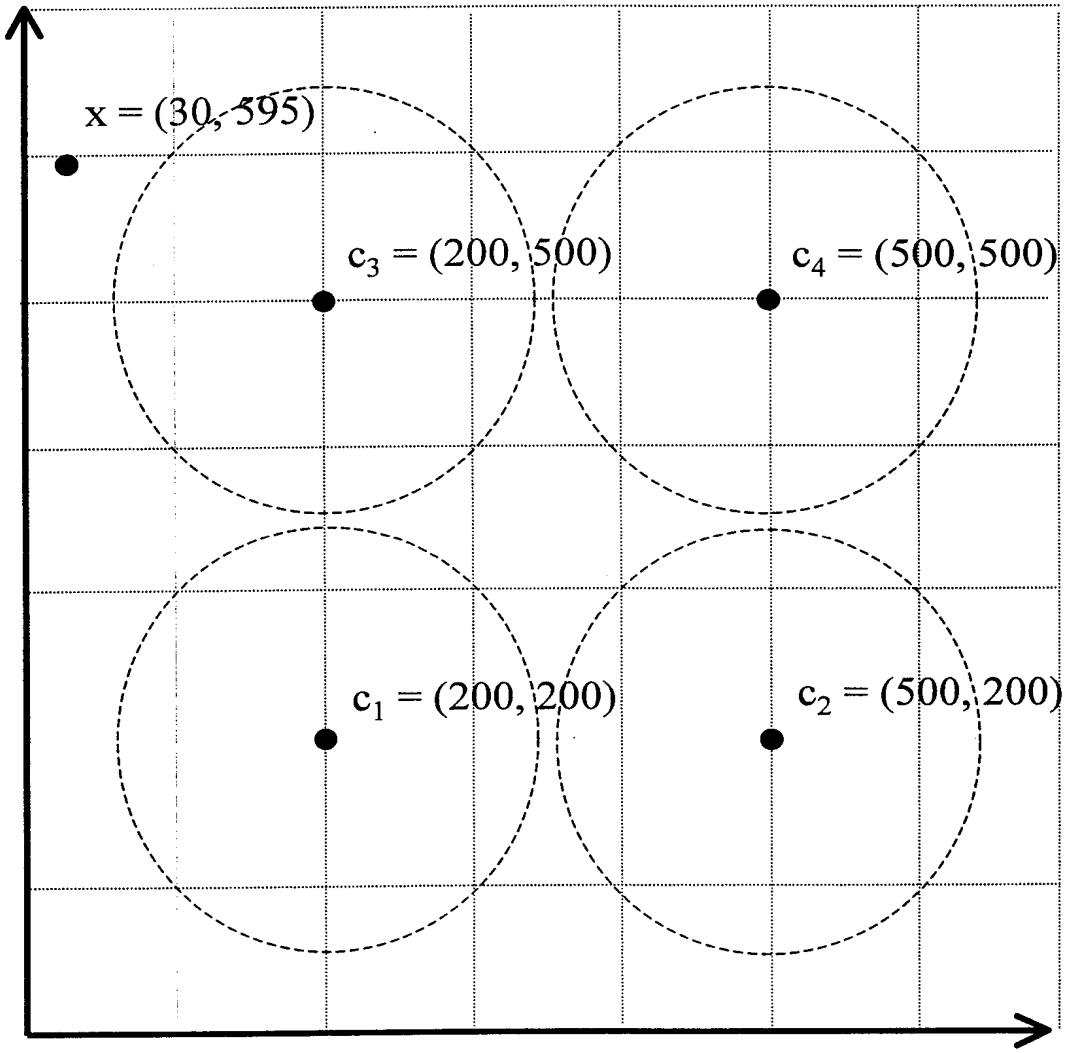


FIG. 3

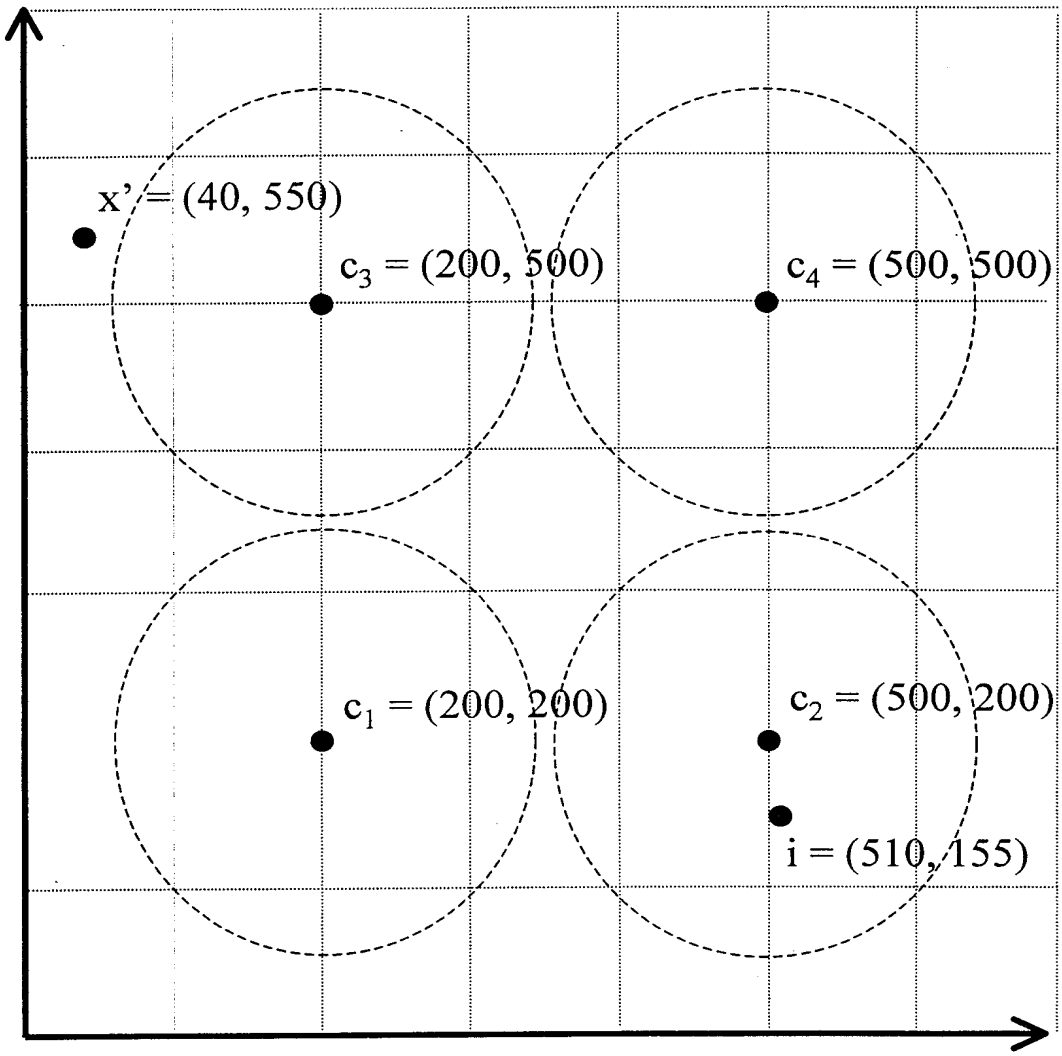


FIG. 4

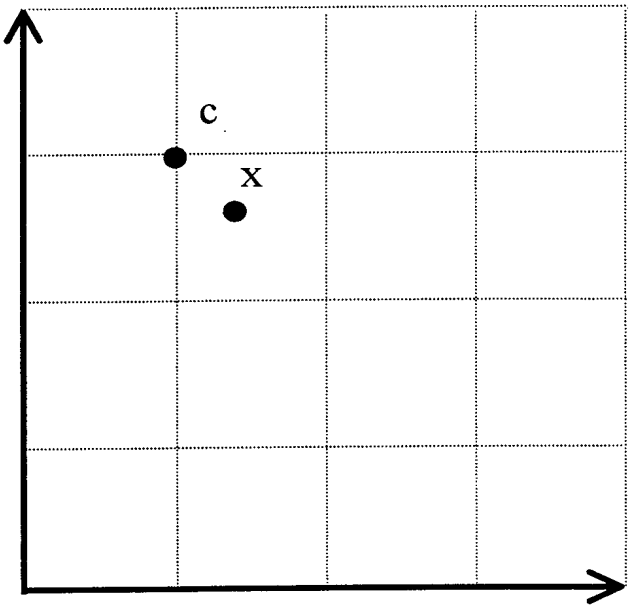


FIG. 5A

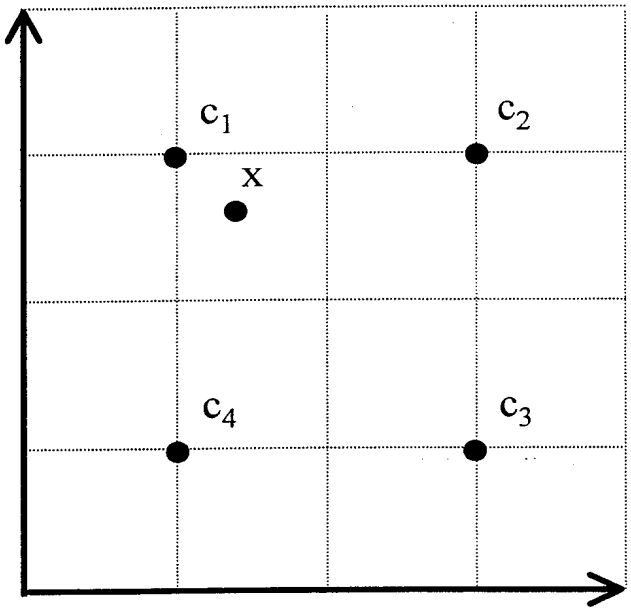


FIG. 5B

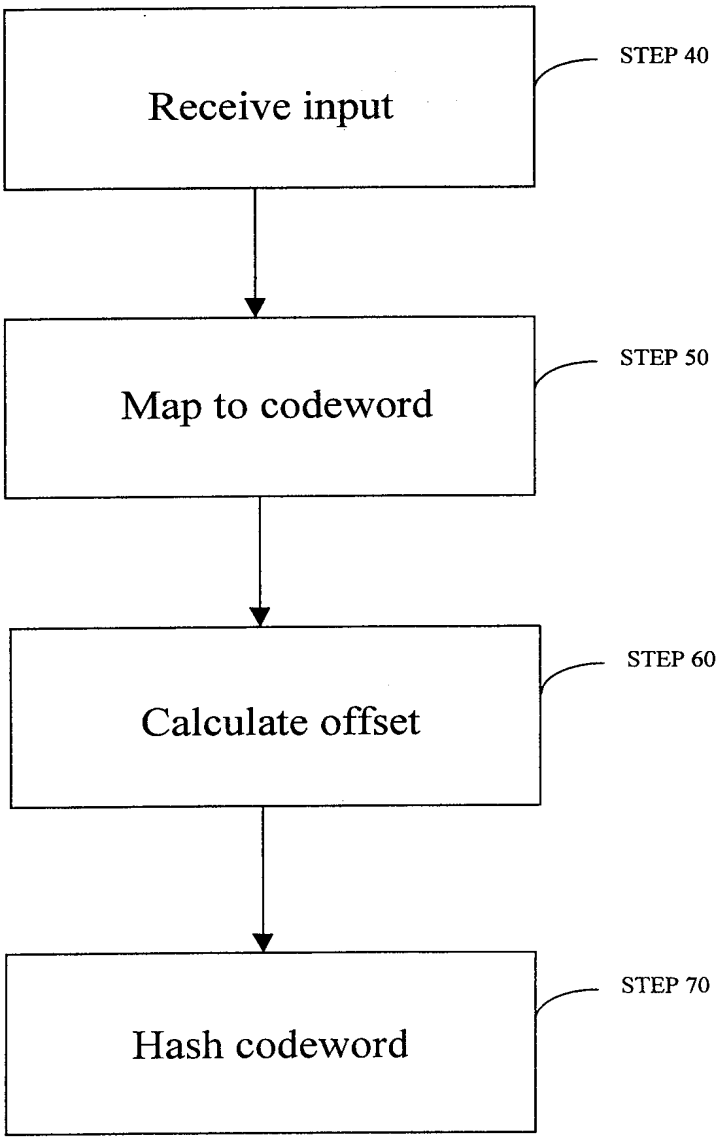

Receive input
STEP 40
Map to codeword
STEP 50
Calculate offset
STEP 60
Hash codeword
STEP 70

FIG. 6

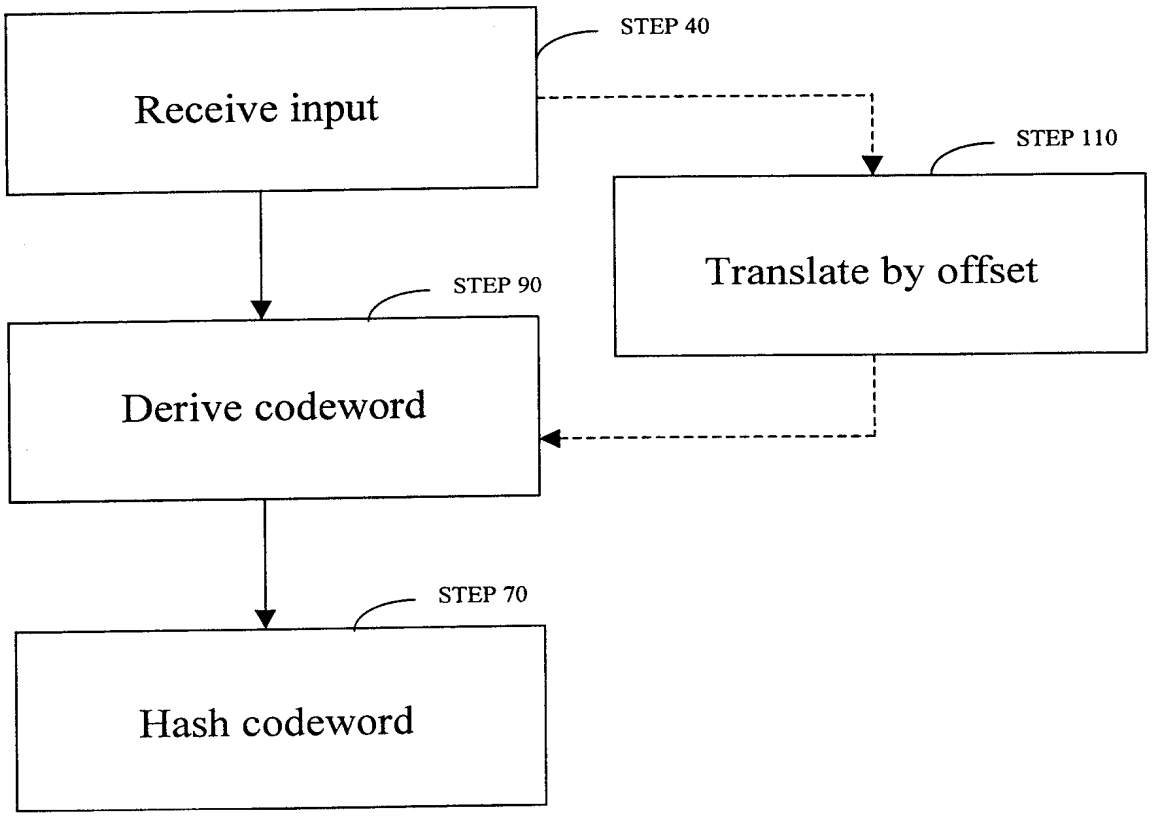

Receive input
STEP 40
Translate by offset
STEP 110
Derive codeword
STEP 90
Hash codeword
STEP 70

FIG. 7

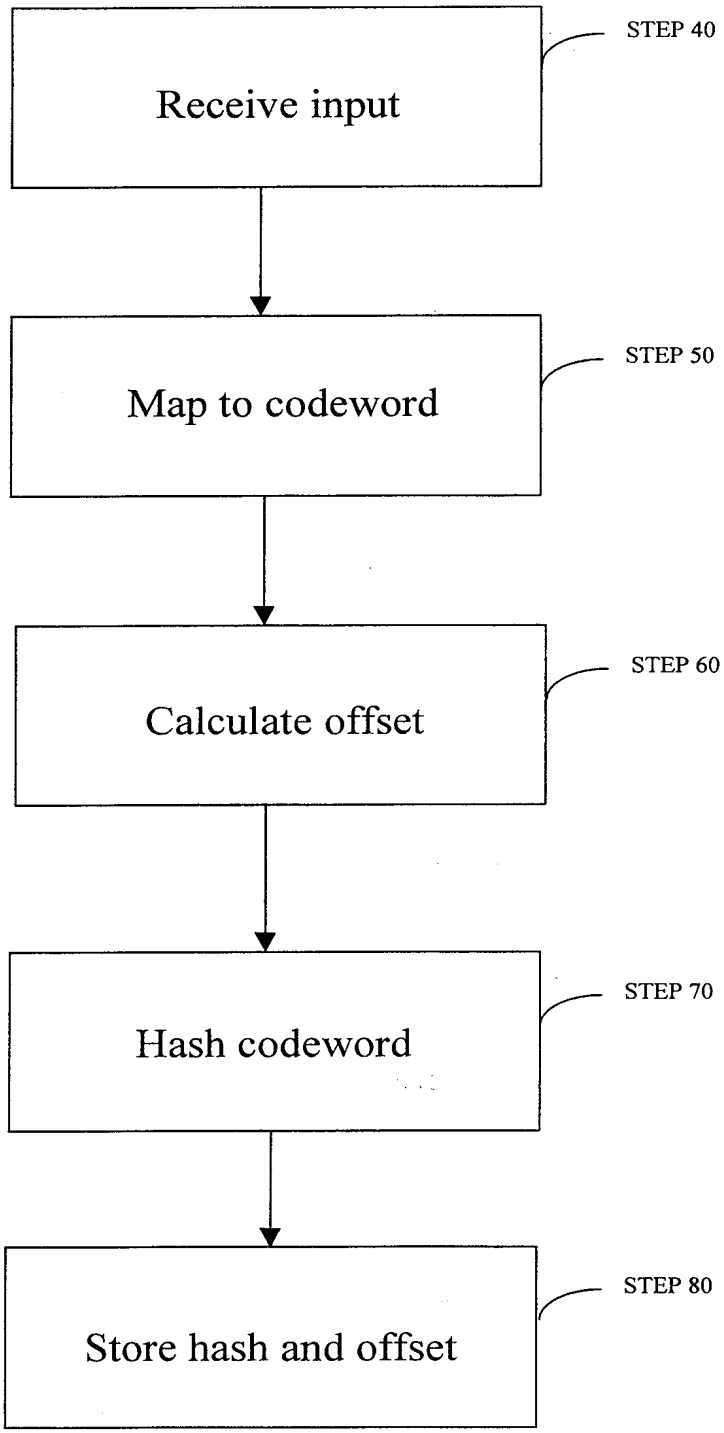

Receive input
STEP 40
Map to codeword
STEP 50
Calculate offset
STEP 60
Hash codeword
STEP 70
Store hash and offset
STEP 80

FIG. 8

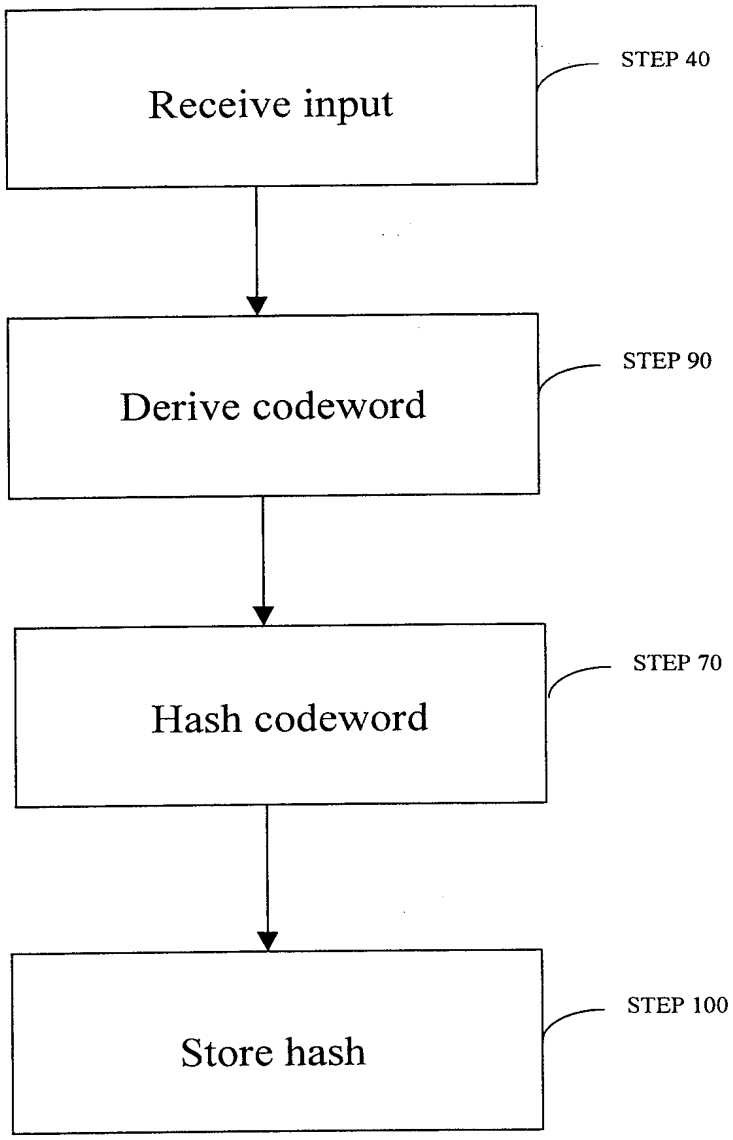

Receive input
STEP 40
Derive codeword
STEP 90
Hash codeword
STEP 70
Store hash
STEP 100

FIG. 9

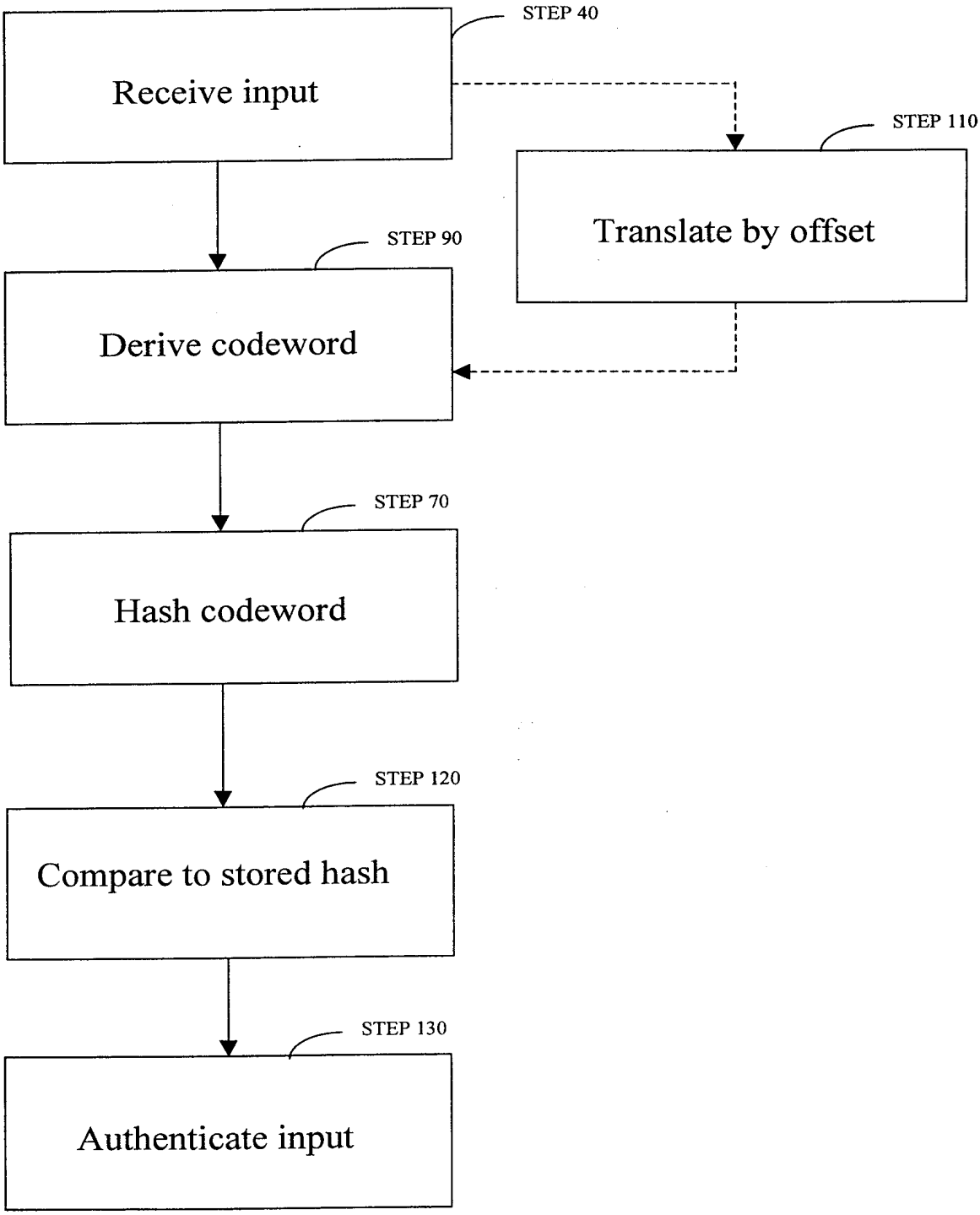

STEP 40
Receive input
STEP 110
Translate by offset
STEP 90
Derive codeword
STEP 70
Hash codeword
STEP 120
Compare to stored hash
STEP 130
Authenticate input

FIG. 10

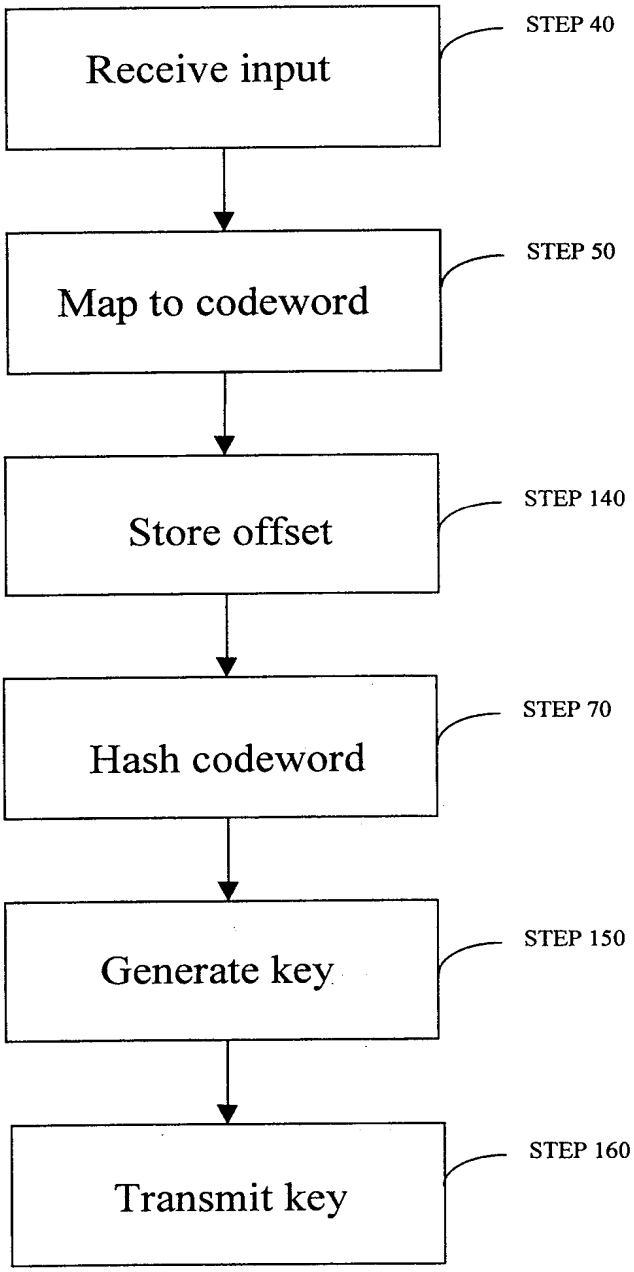

Receive input
STEP 40
Map to codeword
STEP 50
Store offset
STEP 140
Hash codeword
STEP 70
Generate key
STEP 150
Transmit key
STEP 160

FIG. 11

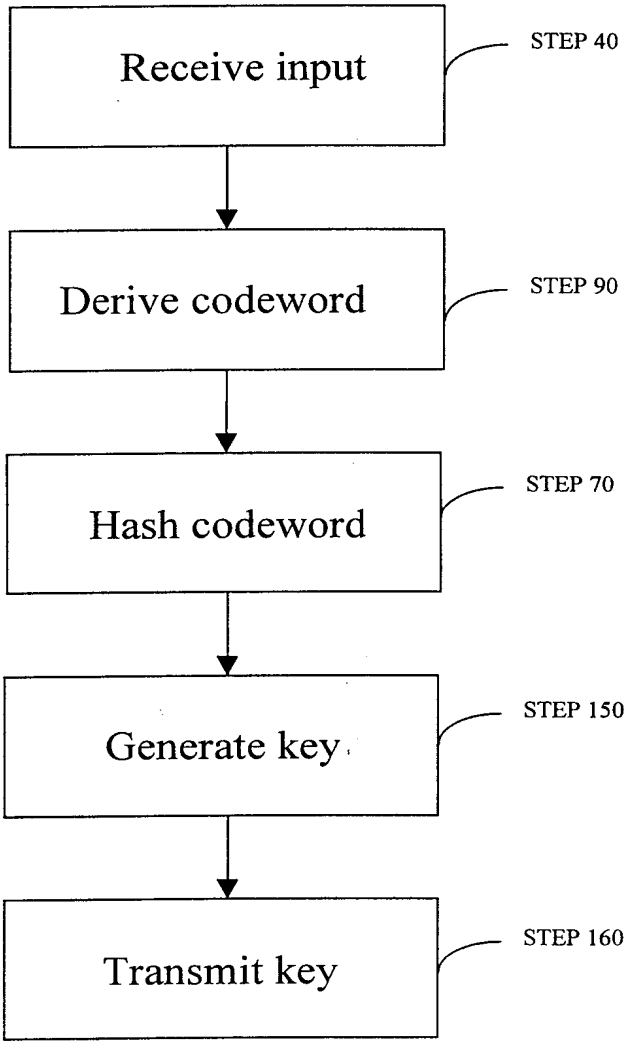

Receive input
STEP 40
Derive codeword
STEP 90
Hash codeword
STEP 70
Generate key
STEP 150
Transmit key
STEP 160

FIG. 12

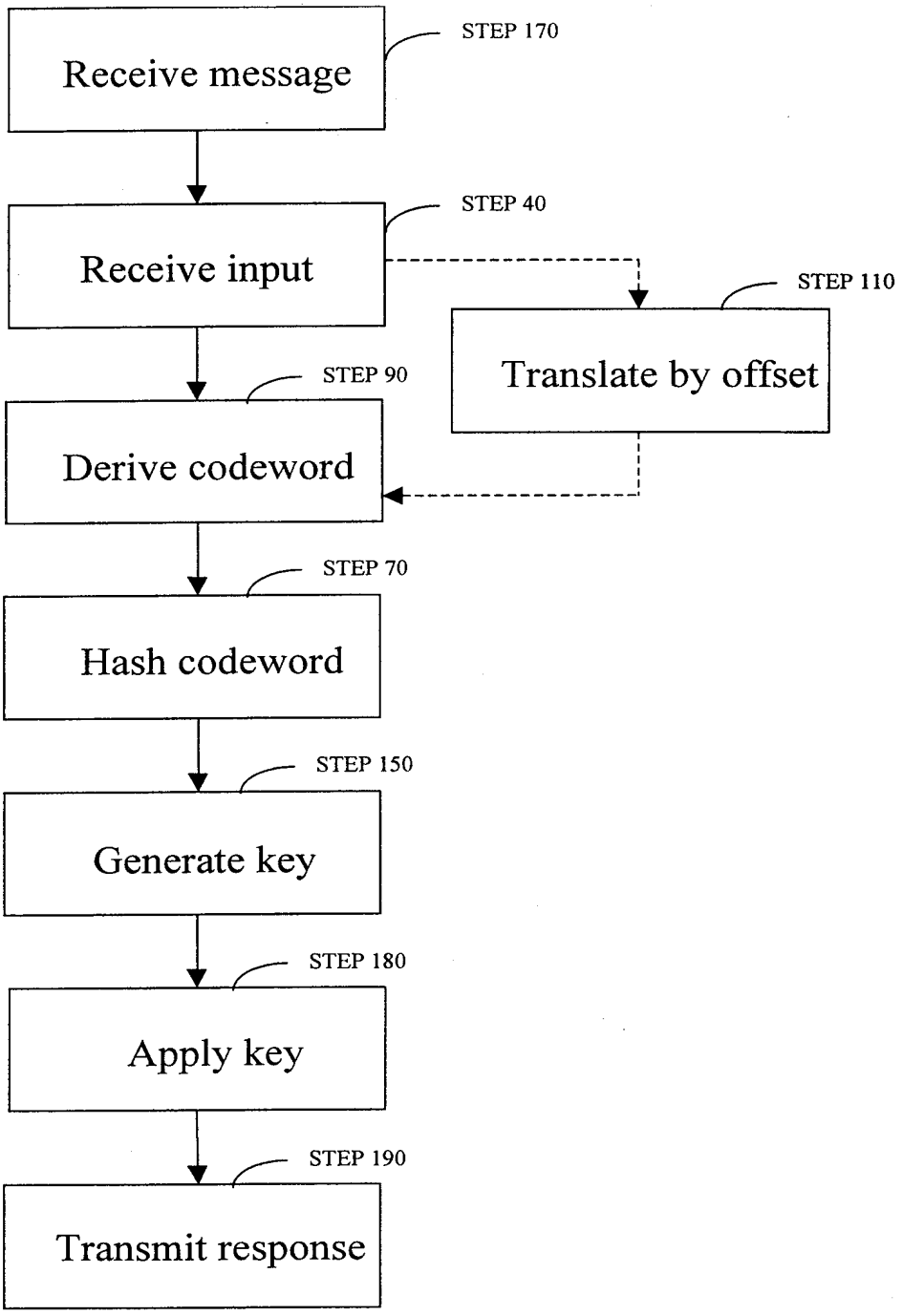

Receive message
STEP 170
Receive input
STEP 40
Translate by offset
STEP 110
Derive codeword
STEP 90
Hash codeword
STEP 70
Generate key
STEP 150
Apply key
STEP 180
Transmit response
STEP 190

FIG. 13

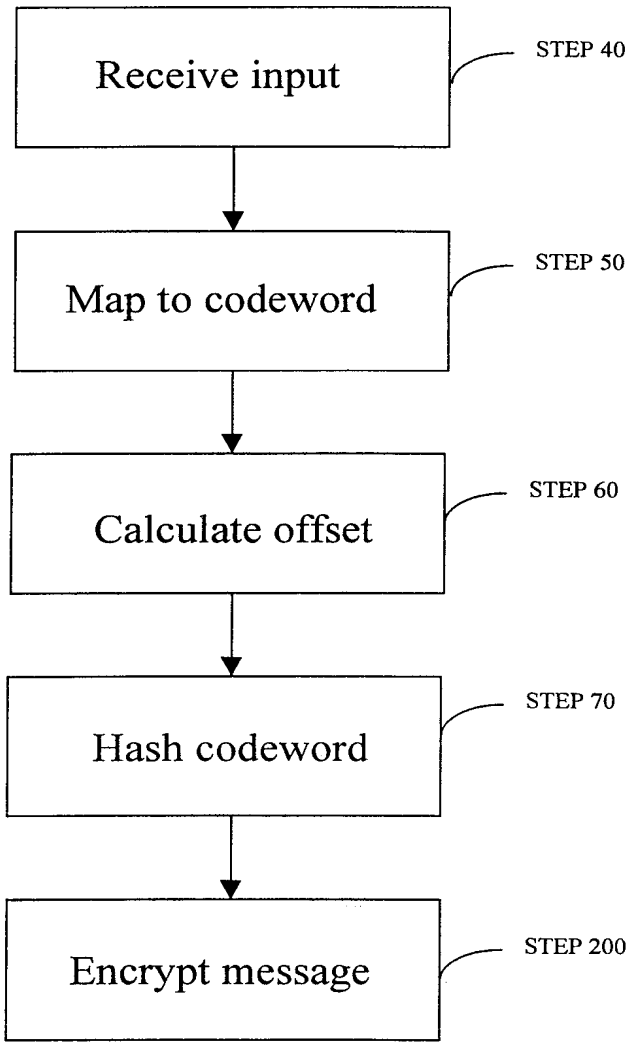

Receive input
STEP 40
Map to codeword
STEP 50
Calculate offset
STEP 60
Hash codeword
STEP 70
Encrypt message
STEP 200

FIG. 14

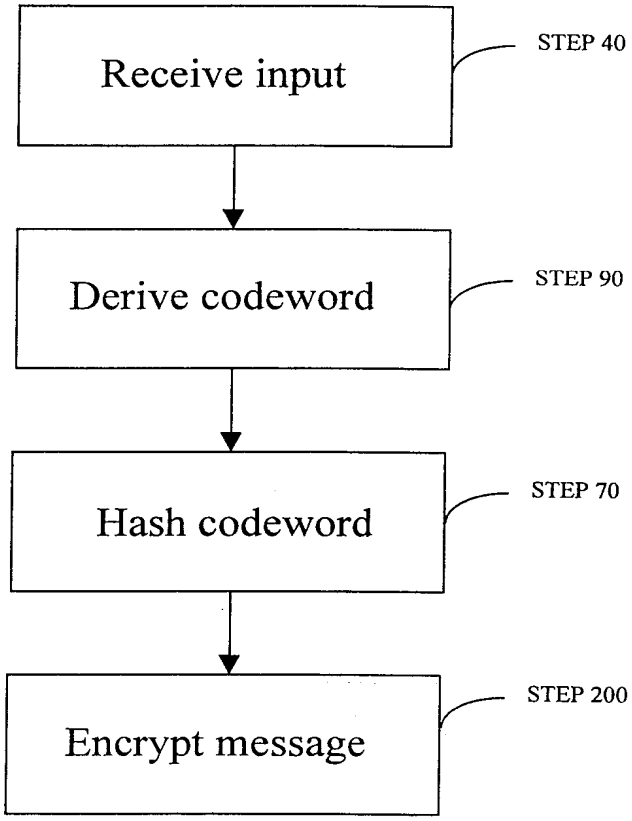

Receive input
STEP 40
Derive codeword
STEP 90
Hash codeword
STEP 70
Encrypt message
STEP 200

FIG. 15

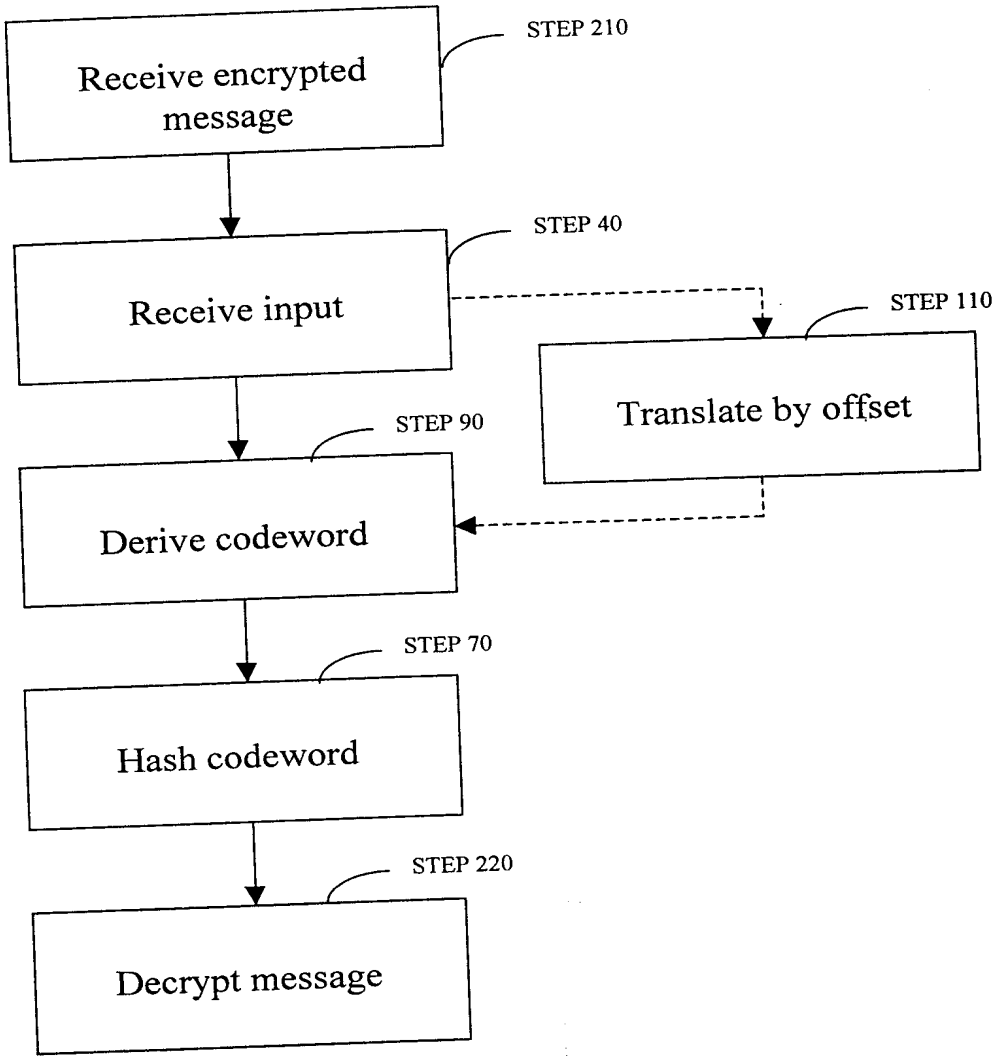

Receive encrypted message
STEP 210
Receive input
STEP 40
Translate by offset
STEP 110
Derive codeword
STEP 90
Hash codeword
STEP 70
Decrypt message
STEP 220

FIG. 16

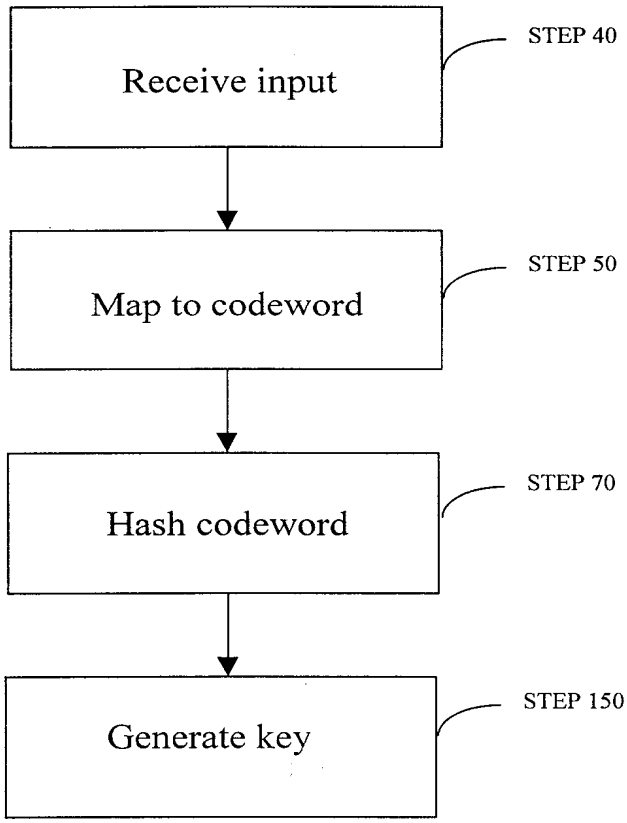

Receive input
STEP 40
Map to codeword
STEP 50
Hash codeword
STEP 70
Generate key
STEP 150

FIG. 17

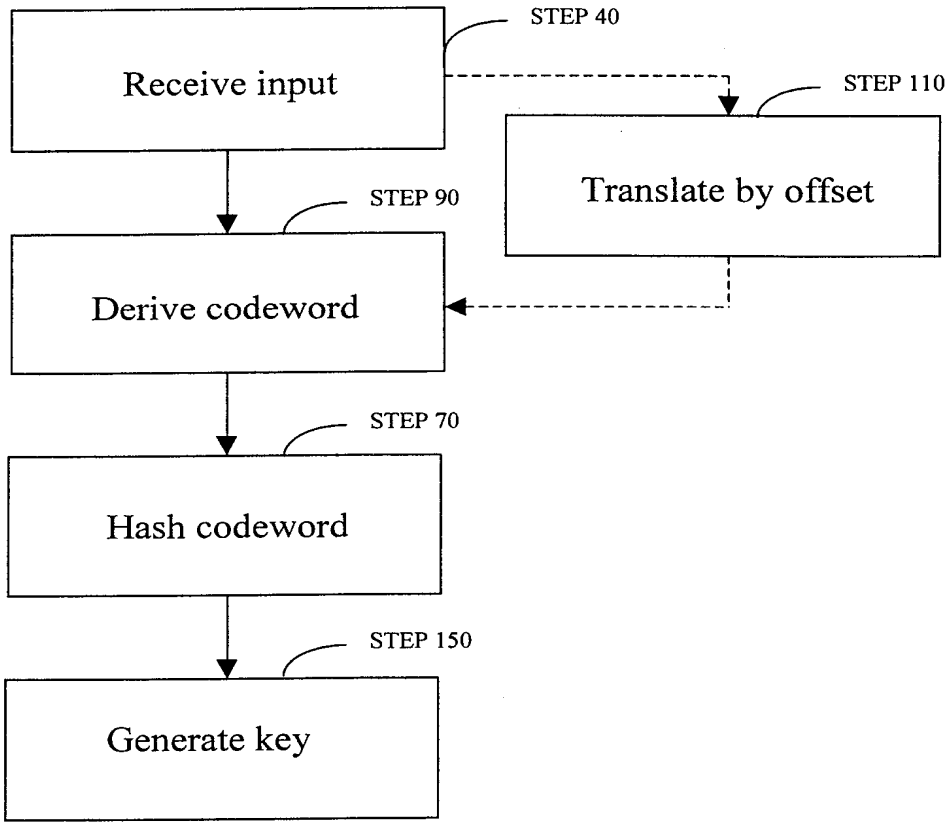

STEP 40
Receive input
STEP 110
Translate by offset
STEP 90
Derive codeword
STEP 70
Hash codeword
STEP 150
Generate key

FIG. 18

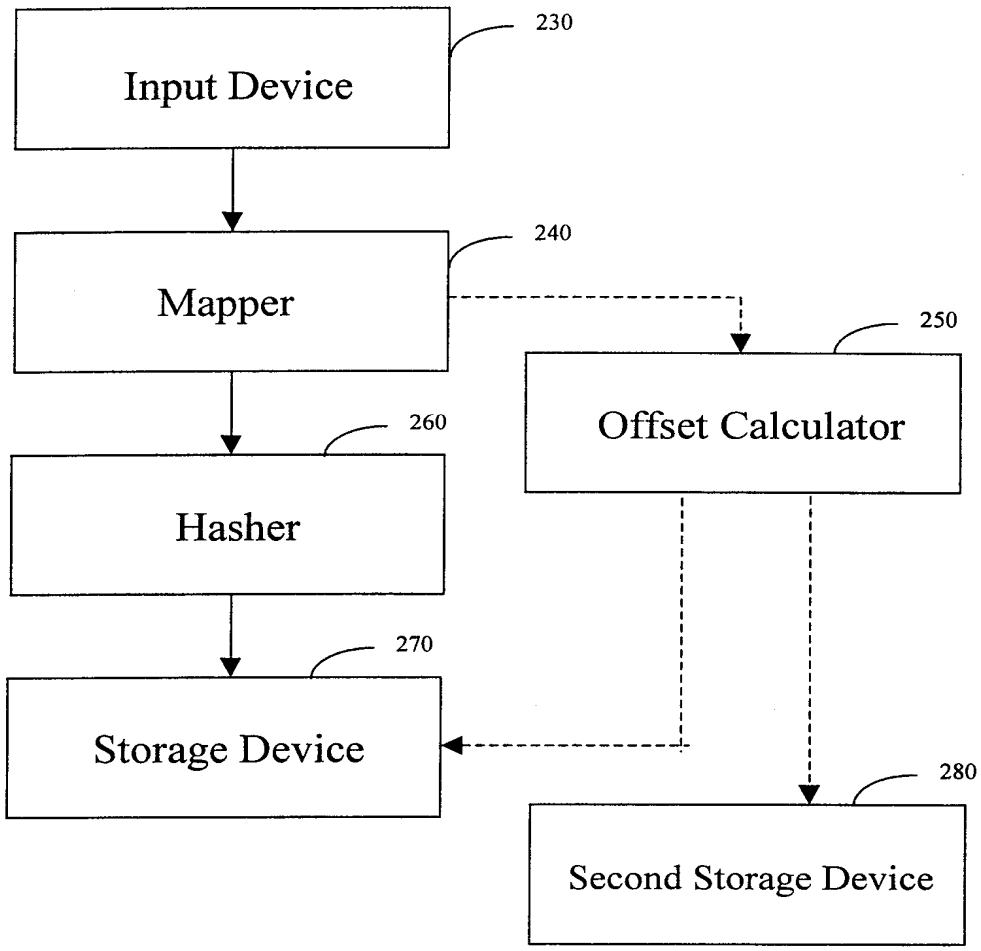

Input Device
230
Mapper
240
Offset Calculator
250
Hasher
260
Storage Device
270
Second Storage Device
280

FIG. 19

INTERNATIONAL SEARCH REPORT

Inter. ɔnal Application No
PCT/US 00/03522

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H03M13/00 H04L9/30

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H03M H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, PAJ, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 97 43736 A (RHOADS GEOFFREY B ;DIGIMARC CORP (US)) 20 November 1997 (1997-11-20) the whole document	1-52
A	FRANKEL Y ET AL: "WITNESS-BASED CRYPTOGRAPHIC PROGRAM CHACKING AND APLLICATIONS" PROCEEDINGS OF THE ANNUAL SYMPOSIUM ON PRINCIPLES OF DISTRIBUTED COMPUTING (PODC),US,NEW YORK, ACM, vol. SYMP. 15, 1996, page 211 XP000681040 ISBN: 0-89791-800-2	

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
11 July 2000	17/07/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Devergranne, C

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Inter onal Application No
PCT/US 00/03522

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	ALABBADI M ET AL: "INTEGRATED SECURITY AND ERROR CONTROL FOR COMMUNICATION NETWORKS USING THE MCELIECE CRYPTOSYSTEM" PROCEEDINGS OF THE INTERNATIONAL CARNAHAN CONFERENCE ON SECURITY TECHNOLOGY: CRIME COUNTERMEASURES,US,NEW YORK, IEEE, vol. -, 1992, pages 172-178, XP000357480 ---	
A	RUBENSTEIN R H: "HARDWARE ACCELERATOR IS PUBLIC'S KEY TO CARD SECURITY" NEW ELECTRONICS,GB,INTERNATIONAL THOMSON PUBLISHING, LONDON, vol. 27, no. 4, 1 April 1994 (1994-04-01), pages 13,15-16, XP000447248 ISSN: 0047-9624 ---	
A	XINMEI W: "DIGITAL SIGNATURE SCHEME BASED ON ERROR-CORRECTING CODES" ELECTRONICS LETTERS,GB,IEE STEVENAGE, vol. 26, no. 13, 21 June 1990 (1990-06-21), pages 898-899, XP000107957 ISSN: 0013-5194 ---	
A	RIEK J R: "OBSERVATIONS ON THE APPLICATION OF ERROR CORRECTING CODES TO PUBLIC KEY ENCRYPTION" PROCEEDINGS OF THE MILITARY COMMUNICATIONS CONFERENCE. (MILCOM),US,NEW YORK, IEEE, vol. -, 1990, pages 1281-1284, XP000221703 ---	
A	RAO T R N ET AL: "PRIVATE-KEY ALGEBRAIC-CODE ENCRYPTIONS" IEEE TRANSACTIONS ON INFORMATION THEORY,US,IEEE INC. NEW YORK, vol. 35, no. 4, 1 July 1989 (1989-07-01), pages 829-833, XP000100912 ISSN: 0018-9448 -----	

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

Inter nal Application No
PCT/US 00/03522

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9743736 A	20-11-1997	US 5862260 A	19-01-1999
		AU 3008697 A	05-12-1997